Uniform Leader Election Protocols for Radio Networks *

Koji Nakano
School of Information Science
Japan Advanced Institute of Science Technology
Tatsunokuchi, Ishikawa 923-1292, JAPAN
knakano@jaist.ac.jp

Stephan Olariu
Department of Computer Science
Old Dominion University
Norfolk, Virginia 23529, USA
olariu@cs.odu.edu

Abstract

A radio network 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. The leader election problem asks to designate one of the station as leader. A leader election protocol is said to be uniform *if in each time slot every station transmits with the same probability.*

In a seminal paper Willard [9] presented a uniform leader election protocol for single-channel single-hop radio stations terminating in $\log\log n + o(\log\log n)$ expected time slots. It was open whether Willard's protocol featured the same time performance with "high probability". We propose a uniform leader election protocol that terminates, with probability exceeding $1 - \frac{1}{f}$ for every $f \geq 1$, in $\log\log n + o(\log\log n) + O(\log f)$ time slots. We also prove that for every $f \in e^{O(n)}$, in order to ensure termination with probability exceeding $1 - \frac{1}{f}$, Willard's protocol must take $\log\log n + \Omega(\sqrt{f})$ time slots. Finally, we provide simulation results that show that our leader election outperforms Willard's leader election protocol in practice.

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 unique radio frequency channel known to all the stations. A 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 [1, 9]. As customary, time is assumed slotted and all transmissions are edge-triggered, that is, take place at time slot boundaries [1, 3]. In a time slot a station can transmit and/or listen to the channel.

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 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.

The problem that we address in this work is the classical *leader election* problem which asks to designate one of the station in the network as *leader*. In other words, after executing the leader election protocol, exactly one station learns that it was elected leader, while the remaining stations learn the identity of the leader.

The leader election problem can be studied in the following three scenarios: *Scenario 1* if the number n of stations is known in advance, *Scenario 2* if the number n of stations is unknown, but an upper bound u on n is known in advance, and *Scenario 3* if 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.

Randomized leader election protocols designed for single-channel, single-hop radio networks work as follows: in each time slot, the stations transmit on the channel with some probability. As we will discuss shortly, this probability may or may not be the same for individual stations. If the status of the channel is SINGLE, the unique station that has transmitted is declared the leader. If the status is not

*Work supported, in part, by the NSF grant CCR-9522093, by ONR grant N00014-97-1-0526, and by a grant from the Telecommunications Advancement Foundation.

SINGLE, the above is repeated until, eventually, a leader is elected. Suppose that a leader election protocol runs for t time slots and a leader has still not been elected at that time. The *history* of a station up to time slot t is captured by

the status of the channel: the status of the channel in each of the t time slots, that is, a sequence of $\{$NULL,COLLISION$\}$ of length t.

transmit/not-transmit: The transmission activity of the station in each of the t time slots, that is, a sequence of $\{$transmit,not-transmit$\}$ of length t.

It should be clear that its history contains all the information that a station can obtain in t time slots. From the perspective of how much of the history information is used, we identify three types of leader election protocols for single-channel, single-hop radio networks: em oblivious if, in time slot i, $(1 \leq i)$, every station transmits with probability p_i and the probability p_i is fixed beforehand and does not depend on the history *uniform* if, in time slot i, $(1 \leq i)$, all the stations transmit with the *same* probability p_i, where p_i is a function of the history of the status of channel in time slots $1, 2, \ldots, i-1$ *non-uniform* if, in each time slot, every station determines its transmission probability depending on its own history.

Several randomized protocols for single-channel, single-hop networks have been presented in the literature. Metcalfe and Boggs [4] presented an oblivious 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. Recently, Nakano and Olariu [6] presented two non-uniform leader election protocols for Scenario 3. The first one terminates, with probability $1 - \frac{1}{n}$, in $O(\log n)$ time slots [1]. The second one terminates with probability $1 - \frac{1}{\log n}$ in $O(\log \log n)$ time slots. The main drawback of these protocols is that the "high probability" expressed by either $1 - \frac{1}{n}$ or $1 - \frac{1}{\log n}$ becomes meaningless for small values of n. For example, the $O(\log \log n)$-time protocol may take a very large number of time slots to terminate. True, this only happens with probability at most $\frac{1}{\log n}$. However, when n is small, this probability is non-negligible.

To address this shortcoming, Nakano and Olariu [7] improved this protocol to terminate, with probability exceeding $1 - \frac{1}{f}$, in $\log \log n + 2.78 \log f + o(\log \log n + \log f)$ time slots. Nakano and Olariu [8] also presented an oblivious leader election protocol for Scenario 3 terminating with probability at least $1 - \frac{1}{f}$, in $O(\min((\log n)^2 + (\log f)^2, f^{\frac{3}{5}} \log n))$ time slots.

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

In a landmark paper, Willard [9] presented a uniform 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, $(0 \leq i \leq \log u)$, satisfying $2^i \leq n < 2^{i+1}$. Once this approximation for n is available, the second stage elects a leader in $O(1)$ expected time slots using the protocol of [4]. Thus, the protocol elects a leader in $\log \log u + O(1)$ expected time slots. Willard [9] 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 [2], which finds an integer i satisfying $2^i \leq n < 2^{i+1}$, bypassing the need for a known upper-bound u on n.

Our first contribution is to propose a uniform leader election protocol terminating, with probability exceeding $1 - \frac{1}{f}$, in $\log \log n + o(\log \log n) + O(\log f)$ time slots. Our uniform leader election features the same performance as the non-uniform leader election protocol of [7] even though all the stations transmit with the same probability in each time slot. This protocol is optimal because, as proved by Willard [9], every uniform leader election protocols needs $\log \log n - O(1)$ expected time slots to terminate and, as we will show later, any uniform protocol that elects a leader with probability at least $1 - \frac{1}{f}$ needs to run for $\log f$ time slots.

Recall that Willard's uniform leader election protocol [9] runs in $\log \log n + o(\log \log n)$ expected time. Since our uniform leader election protocol runs in $\log \log n + o(\log \log n)$ expected time slots, it features the same performance as Willard's protocol in terms of the *expected* number of time slots. However, the distribution of the time slots is different in the two protocols. In order to show this fact, we prove that with probability at least $1 - \frac{1}{f}$ Willard's protocol has to run for at least $\log \log n + \Omega(\sqrt{f})$ time slots to elect a leader. Thus, Willard's protocol stands a much larger chance than our protocol to run for a long time before electing a leader.

Further, as we are going to show, $\Omega(\sqrt{f})$ is a dominant factor for some applications. Suppose that n stations are partitioned into $\sqrt{n}$ clusters of $\sqrt{n}$ stations each. Let us consider a task involving the following two steps:

Step 1 elect a leader in each cluster;

Step 2 elect a leader among the leaders elected in Step 1.

Note that, all of the $\sqrt{n}$ leaders must be elected in Step 1 before starting Step 2. Using our leader election protocol, Step 1 terminates, with probability at least $1 - \frac{1}{f}$, in $\log \log \sqrt{n} + o(\log \log \sqrt{n}) + O(\log(f\sqrt{n})) = O(\log n + \log f)$ time slots.

Step 2 takes $\log\log\sqrt{n} + o(\log\log\sqrt{n}) + O(\log f) = \log\log n + o(\log\log n) + O(\log f)$ time slots. Thus, using our leader election protocol, the task can be completed in $O(\log n)$ expected time slots. On the other hand, using Willard's leader election protocol, the expected time slots to complete this task is much larger, even if Willard's leader election protocol runs for $\log\log n + o(\log\log n) + O(\sqrt{f})$. Step 1 takes $\log\log\sqrt{n} + o(\log\log\sqrt{n}) + O(\sqrt{f\sqrt{n}}) = O(n^{\frac{1}{4}} + \sqrt{f})$ time slots. Step 2 takes $\log\log n + o(\log\log n) + O(\sqrt{f})$ time slots. Consequently, using Willard's protocol, the task is completed, with probability $1 - \frac{1}{f}$, in $O(n^{\frac{1}{4}} + \sqrt{f})$ time slots and thus, in $O(n^{\frac{1}{4}})$ expected time slots. Arguably, Willard's protocol is much slower than our protocol to complete this task.

We also provide simulation results that show that our protocol is practically fast and that Willard's uniform leader election protocol is very slow with some probability. More precisely, in 1,000,000 simulations for various values of n up to 100,000,000, our protocol never required more than 52 time slots, while Willard's protocol needed more than 1,600 time slots in the worst case.

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 [5].

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, $(0 \leq r \leq n)$,

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

Further, the expected value of X is given by

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

To analyze the tail of the binomial distribution, we shall make use of the following estimates, commonly referred to as *Chernoff bounds* [5]:

$$\Pr[X > (1+\delta)E[X]] \;<\; \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{E[X]} \quad (0 \leq \delta) \quad (1)$$

$$\Pr[X > (1+\epsilon)E[X]] \;<\; e^{-\frac{\epsilon^2}{3}E[X]} \quad (0 \leq \epsilon \leq 1) \quad (2)$$

$$\Pr[X < (1-\epsilon)E[X]] \;<\; e^{-\frac{\epsilon^2}{2}E[X]} \quad (0 \leq \epsilon \leq 1). \quad (3)$$

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

$$\Pr[X \geq c \cdot E[X]] \;\leq\; \frac{1}{c} \qquad \text{for all } c \geq 1. \quad (4)$$

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

Lemma 2.1 *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, $E[X] \leq \int_0^1 T(F)dF$.*

3 Uniform leader election protocols

The main purpose of this section is to develop a uniform leader election protocol 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. We begin by presenting a very simple protocol that is the workhorse of all subsequent leader election protocols.

Protocol `Broadcast(`p`)`
  every station transmits with probability $\frac{1}{2^p}$;
  if the status of the channel is SINGLE **then**
    the unique station that has transmitted becomes the leader and all stations exit the (main) protocol

---

In Subsection 3.1 we begin by exhibiting a first uniform leader election protocol terminating, with probability exceeding $1 - \frac{1}{f}$, in $2\log\log n + O(\log f) + o(\log\log n)$ time slots. In Subsection 3.2 we show how this protocol can be modified to run in $\log\log n + O(\log f) + o(\log\log n)$ time slots.

3.1 A uniform leader election protocol running in $2\log\log n$ time slots

In outline, our leader election protocol proceeds in three phases.
In Phase 1 the calls `Broadcast(`2^0`)`, `Broadcast(`2^1`)`, `Broadcast(`2^2`)`, ..., `Broadcast(`2^t`)` are performed until, for the first time, the status of the channel is NULL in `Broadcast(`2^t`)`. At this point Phase 2 begins. Phase 2 executes a variant of binary search on the interval $[0, 2^t]$ using the protocol `Broadcast` as follows:

- First, `Broadcast(`$\frac{2^t}{2}$`)` is executed. If the status of the channel is SINGLE then the unique station that has transmitted becomes the leader.
- If the status of channel is NULL then binary search is performed on the interval $[0, \frac{2^t}{2}]$, that is, `Broadcast(`$\frac{2^t}{4}$`)` is executed.

- If the status of channel is COLLISION then binary search is performed on the interval $[\frac{2^t}{2}, 2^t]$, that is, `Broadcast(`$\frac{3}{4} \cdot 2^t$`)` is executed.

This procedure is repeated until, at some point, binary search cannot further split an interval. Let u be the integer such that the last call of Phase 2 is `Broadcast(`u`)`. Phase 3 repeats the call `Broadcast(`u`)` until, eventually, the status of the channel is SINGLE, at which point a leader has been elected. It is important to note that the value of u is continuously adjusted in Phase 3 as follows: if the status of the channel is NULL, then it is likely that 2^u is larger than n. Thus, u is decreased by one. By the same reasoning, if the status of the channel is COLLISION, u is increased by one.

With this preamble out of the way, we are now in a position to spell out the details of our uniform leader election protocol.

Protocol `Uniform-election`
Phase 1:
  $i \leftarrow -1$;
  repeat
    $i \leftarrow i + 1$;
    `Broadcast(`2^i`)`
  until the status of the channel is NULL;
Phase 2:
  $l \leftarrow 0$; $u \leftarrow 2^i$;
  while $l + 1 < u$ **do**
    $m \leftarrow \lceil \frac{l+u}{2} \rceil$;
    `Broadcast(`m`);`
    if the status of channel is NULL **then**
      $u \leftarrow m$
    else
      $l \leftarrow m$
  endwhile
Phase 3:
  repeat
    `Broadcast(`u`);`
    if the status of channel is NULL **then**
      $u \leftarrow \max(u - 1, 0)$
    else
      $u \leftarrow u + 1$
  forever

We now turn to the task of evaluating the number of time slots it takes the protocol to terminate. In Phase 1, once the status of the channel is NULL the protocol exits the `repeat-until` loop. Thus, there exist an integer t such that the status of the channel is:

- SINGLE or COLLISION in the calls `Broadcast(`2^0`)`, `Broadcast(`2^1`)`, `Broadcast(`2^2`)`, ..., `Broadcast(`2^{t-1}`)`, and
- NULL in `Broadcast (`2^t`)`.

Let $f \geq 1$ be arbitrary and write

$$s = \lceil \log \log(4nf) \rceil. \tag{5}$$

To motivate the choice of s in (5) we show that with probability exceeding $1 - \frac{1}{4f}$, s provides an upper bound on t. Let X be the random variable denoting the number of stations that transmit in `Broadcast(`2^s`)`. The probability that a particular station is transmitting in the call `Broadcast(`2^s`)` is less than $\frac{1}{2^{2^s}}$. Thus, the expected value $E[X]$ of X is upper-bounded by

$$E[X] < \frac{n}{2^{2^s}} \leq \frac{n}{4nf} = \frac{1}{4f}. \tag{6}$$

Using the Markov inequality (4) and (6) combined, we can write

$$\Pr[X \geq 1] < \Pr[X \geq 4fE[X]] \leq \frac{1}{4f}. \tag{7}$$

Equation (7) implies that with probability exceeding $1 - \frac{1}{4f}$, the status of the channel at the end of the call `Broadcast(`2^s`)` is NULL confirming that

$$t \leq s \text{ holds with probability exceeding } 1 - \frac{1}{4f}. \tag{8}$$

Thus, with probability exceeding $1 - \frac{1}{4f}$, Phase 1 terminates in

$$t+1 \leq s+1 = \lceil \log \log(4nf) \rceil + 1 = \log \log n + O(\log \log f)$$

time slots. Since Phase 2 terminates in at most $s + 1 = \log \log n + O(\log \log f)$ time slots, we have proved the following result.

Lemma 3.1 *With probability exceeding $1 - \frac{1}{4f}$, Phase 1 and Phase 2 combined take at most $2 \log \log n + O(\log \log f)$ time slots.*

Our next goal is to evaluate the value of u at the end of Phase 2. For this purpose, we say that the call `Broadcast(`m`)` executed in Phase 2 *fails*

- if $n \leq \frac{2^m}{4(s+1)f}$ and yet the status of the channel is COLLISION, or
- if $n \geq 2^m \cdot \ln(4(s + 1)f)$ and yet the status of the channel is NULL.

We are interested in evaluating the probability that `Broadcast(`m`)` fails. Let Y be the random variable denoting the number of stations transmitting in the call `Broadcast(`m`)`. First, if $n \leq \frac{2^m}{4(s+1)f}$, then $E[Y] =$

$\frac{n}{2^m} \leq \frac{1}{4(s+1)f}$ holds. By using the Markov inequality (4), we have

$$\Pr[Y > 1] \quad \leq \quad \Pr[Y > 4(s+1)f \cdot E[Y]] < \frac{1}{4(s+1)f}.$$

It follows that the status of the channel is COLLISION with probability at most $\frac{1}{4(s+1)f}$.

Next, suppose that $n \geq 2^m \cdot \ln(4(s+1)f)$ holds. The status of the channel is NULL with probability at most

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

Clearly, in either case, the probability that the call `Broadcast(`m`)` *fails* is at most $\frac{1}{4(s+1)f}$. Importantly, this probability is independent of m. Since the protocol `Broadcast` is called at most $s+1$ times in Phase 2, the probability that *none* of these calls fails is at least $1-\frac{1}{4f}$. On the other hand, recall that the probability that `Broad-cast` is called at most $s+1$ times exceeds $1-\frac{1}{4f}$. Now a simple argument shows that the probability that Phase 2 involves at most $s+1$ calls to `Broadcast` and that none of these calls fail exceeds $1-\frac{1}{2f}$. Thus, we have proved the following result.

Lemma 3.2 *With probability exceeding* $1-\frac{1}{2f}$*, when Phase 2 terminates* u *satisfies the double inequality* $\frac{n}{\ln(4(s+1)f)} \leq 2^u \leq 4(s+1)fn$*.*

Finally, we are interested in getting a handle on the number of time slots involved in Phase 3. For this purpose, let v, $(1 \leq v)$, be the integer satisfying the double inequality

$$2^{v-1} < n \leq 2^v. \tag{9}$$

A generic call `Broadcast(`u`)` performed in Phase 3 is said to *fail to decrease* if $u \geq v+2$ and yet the status of the channel is COLLISION, *succeed to decrease* if $u \geq v+3$ and yet the status of the channel is NULL, *fail to increase* if $u \leq v-2$ and yet the status of the channel is NULL, *succeed to increase* if $u \leq v-3$ and yet the status of the channel is COLLISION. *good*, otherwise.

More generally, we say that the call `Broadcast(`u`)` *fails* if it fails either to increase or to decrease; similarly, the call `Broadcast(`u`)` is said to *succeed* if it succeeds either to increase or to decrease. The motivation for this terminology comes from the observation that if `Broadcast(`u`)` succeeds then, u is updated so that u approaches v.

Assume that the call `Broadcast(`u`)` is good. Clearly, if such is the case, the double inequality $v-2 \leq u \leq v+2$ holds at the beginning of the call. The status of the channel in `Broadcast(`u`)` is SINGLE with probability at least

$$\begin{aligned} &\binom{n}{1}\frac{1}{2^u}(1-\frac{1}{2^u})^{n-1} \\ &> \frac{n}{2^u}e^{-\frac{n}{2^u}} \\ &> \min(\frac{2^{u+2}}{2^u}e^{-\frac{2^{u+2}}{2^u}}, \frac{2^{u-3}}{2^u}e^{-\frac{2^{u-3}}{2^u}}) \\ &> \min(\frac{4}{e^4}, \frac{1}{8e^{\frac{1}{8}}}) = \frac{4}{e^4}. \end{aligned}$$

We note that this probability is independent of u. Thus, if a good call is executed $\frac{e^4}{4}\ln(4f)$ times, a leader is elected with probability at least

$$1-(1-\frac{4}{e^4})^{\frac{e^4}{4}\ln(4f)} > 1-e^{-\ln(4f)} = 1-\frac{1}{4f}.$$

As we are about to show, good calls occur quite frequently in Phase 3. To this end, we prove an upper bound on the probability that the call `Broadcast(`u`)` fails. Let Z denote the number of stations that transmit in `Broadcast(`u`)`. Clearly, $E[Z] = \frac{n}{2^u}$. Thus, if $u \geq v+2$ then the call `Broadcast(`u`)` fails to decrease with probability at most

$$\begin{aligned} \Pr[Z > 1] &= \Pr[Z > \frac{2^u}{n}E[Z]] \\ &< \Pr[Z > \frac{2^u}{2^v}E[Z]] \qquad \text{(from } n \leq 2^v\text{)} \\ &< \Pr[Z > 4E[Z]] \qquad \text{(from } u \geq v+2\text{)} \\ &< \frac{1}{4} \qquad \text{(by Markov's inequality (4)).} \end{aligned}$$

On the other hand, if $u \leq v-2$ then the probability that `Broadcast(`u`)` fails to increase is at most

$$\begin{aligned} \Pr[Z=0] &= (1-\frac{1}{2^u})^n \\ &< e^{-\frac{n}{2^u}} \\ &< e^{-\frac{2^{v-1}}{2^u}} \qquad \text{(from } 2^{v-1} < n\text{)} \\ &< e^{-2} \qquad \text{(from } u \leq v-2\text{)} \\ &< \frac{1}{4}. \end{aligned}$$

Therefore, the call `Broadcast(`u`)` fails with probability at most $\frac{1}{4}$.

Suppose that `Broadcast(`u`)` is executed $\frac{8}{3}e^4(\ln(4f)+\log\log\log n)$ times in Phase 3 and let N_s, N_f, and N_g be, respectively, the number of times `Broadcast(`u`)` succeeds, fails, and is good among these $\frac{8}{3}e^4(\ln(4f)+\log\log\log n)$ calls. Clearly,

$$N_s + N_f + N_g \quad = \quad 4e^4(\ln f + \log\log\log n). \tag{10}$$

If at the end of Phase 2 u satisfies the double inequality of Lemma 3.2, we have

$$\begin{aligned} u &\geq \log(\frac{n}{4(s+1)f}) \\ &> \log n - \log(s+1) - \log\log f - 2 \\ &> v - \log\log f - \log\log\log n - \log\log\log f - 4, \end{aligned}$$

and, similarly,

$$\begin{aligned} u &\leq \log(n(\ln(4(s+1)f))) \\ &< \log n + \log\ln(s+1) + \log\ln f + 2 \\ &< v + \log\log f + \log\log\log\log n \\ &\quad + \log\log\log\log f + 3. \end{aligned}$$

Thus, we have,

$$|u - v| < 2\log\log f + \log\log\log n + 4. \quad (11)$$

We note that if $|u - v| \leq 2$ holds at the end of Phase 2, then $N_s \leq N_f$. By the same reasoning, it is easy to see that if (11) holds at the end of Phase 2, we have

$$N_s \quad < \quad N_f + 2\log\log f + \log\log\log n + 2. \quad (12)$$

Since a particular call `Broadcast(u)` fails with probability at most $\frac{1}{4}$, we have

$$E[N_f] \leq \frac{2e^4}{3}(\ln(4f) + \log\log\log n).$$

Thus, the probability that there are more than $e^4(\ln(4f) + \log\log\log n)$ calls fails is at most

$$\begin{aligned} &\Pr[N_f > e^4(\ln(4f) + \log\log\log n)] \\ &< \Pr[N_f > (1 + \frac{1}{2})E[N_f]] < e^{-\frac{1}{2^2 \cdot 3}E[N_f]} \\ &< e^{-\frac{e^4}{24}(\ln(4f) + \log\log\log n)} < \frac{1}{4f}. \end{aligned}$$

Suppose that $N_f \leq e^4(\log(4f) + \log\log\log n)$ is satisfied. Then, we have

$$\begin{aligned} N_g &= \frac{8}{3}e^4(\ln(4f) + \log\log\log n) - (N_s + N_f) \\ &\geq \frac{8}{3}e^4(\ln(4f) + \log\log\log n) - 2N_f \\ &\quad -(2\log\log f + \log\log\log n + 2) \\ &> \frac{e^4}{2}\ln(4f). \end{aligned}$$

Therefore, with probability at least $1 - \frac{1}{4f}$, among $\frac{e^4}{2}(\log(4f) + \log\log\log n)$ calls `Broadcast(u)` there are at least $\frac{2}{3}e^4\ln(4f)$ good ones. It follows that if at the end of Phase 2 u satisfies the double in equality in Lemma 3.2, then with probability $1 - \frac{1}{2f}$, Phase 3 terminates in at most $\frac{e^4}{2}(\log f + \log\log\log n)$ time slots. To summarize, we have proved the following result.

Lemma 3.3 *Protocol* `Uniform-election` *terminates, with probability at least* $1 - \frac{1}{f}$*, in at most* $2\log\log n + o(\log\log n) + O(\log f)$ *time slots.*

3.2 Uniform leader electing protocol running in $\log\log n$ time slots

The main goal of this subsection is to outline the changes that will make protocol `Uniform-election` terminate, with probability exceeding $1 - \frac{1}{f}$, in $\log\log n + o(\log\log n) + O(\log f)$ time slots.

In Phase 1 the calls `Broadcast(`2^{0^2}`)`, `Broadcast(`2^{1^2}`)`, `Broadcast(`2^{2^2}`)`, ..., `Broadcast(`2^{t^2}`)` are performed until, for the first time, the status of the channel is NULL in `Broadcast(`2^{t^2}`)`. Phase 2 performs binary search on $[0, 2^{t^2}]$ using `Broadcast` as discussed in Subsection 3.1. The reader should be in a position to confirm that $t \leq \lceil\sqrt{\log\log(4nf)}\rceil$ is satisfied with probability at least $1 - \frac{1}{f}$, for any $f \geq 1$. Thus, Phase 1 terminates in $\lceil\sqrt{\log\log(4nf)}\rceil + 1$ time slots, while Phase 2 terminates in $(\lceil\sqrt{\log\log(4nf)}\rceil + 1)^2$ time slots. Therefore, with probability at least $1 - \frac{1}{f}$, Phase 1 and 2 combined terminate in $\log\log n + o(\log\log n) + O(\log\log f)$ time slots. Thus, we have the following result.

Theorem 3.4 *There exists a uniform leader election protocol that terminates, with probability at least* $1 - \frac{1}{f}$*, in* $\log\log n + o(\log\log n) + O(\log f)$ *time slots, for every* $f \geq 1$.

It is worth noting that by Lemma 2.1 and Theorem 3.4 combined, our uniform leader election protocol terminates in $\log\log n + o(\log\log n)$ expected time slots.

4 Willard's uniform leader election protocol

The main goal of this section is to take a very close look at the performance of Willard's uniform leader election protocol [9]. As it turns out, our uniform leader election protocol presented in Section 3 is very similar to the one in [9]. Both our protocol and Willard's consists of three phases. Phases 1 and 2 are the same for the two protocols. In Willard's protocol, once u is determined at the end of Phase 2, `Broadcast(u)` is repeated until a leader is elected. Recall that Phase 3 of our protocol keeps adjusting the value of u in each time slot, depending on the status of the channel. On the other hand, Phase 3 of Willard's protocol does not changes the value of u.

Quite surprisingly, both our and Willard's protocol feature the exact same expected-time performance, terminating in $\log\log n + o(\log\log n)$ expected time slots. However, as we are going to show, in order to elect a leader with probability at least $1 - \frac{1}{f}$ for every f such that $f \le e^{\frac{n+1}{2}}$, Willard's protocol must run for at least $\log\log n + \Omega(\sqrt{f})$ time slots.

Let us consider Willard's version of `Uniform-election`. In other words, Phase 3 repeats the call `Broadcast(`u`)` without changing the value of u. Suppose that $n = 2^{2^w}$ for some integer w. Recall that Phase 1 involves the calls `Broadcast(`2^{0^2}`)`, `Broadcast(`2^{1^2}`)`, `Broadcast(`2^{2^2}`)`, ..., until the status of channel is NULL for the first time.

We assume, without loss of generality, that Phase 1 terminates as a result of the fact that the status of the channel is NULL in the call `Broadcast(`2^{2^w}`)` and prove that Phase 3 takes $\Omega(\sqrt{f})$ time slots for this case. The proof of the case where Phase 1 terminates earlier or later is similar and therefore omitted.

After Phase 2, if $u = 2^w$, then each of the calls in Phase 3 elects a leader with probability at least

$$\binom{n}{1}\left(1-\frac{1}{2^m}\right)^{n-1}\frac{1}{2^m} > e^{-\frac{n}{2^m}}\frac{n}{2^m} > \frac{1}{2\sqrt{e}}.$$

It follows that Phase 2 elects a leader in $\frac{1}{\log(1-\frac{1}{2\sqrt{e}})}\log f \approx 1.92\log f$ time slots with probability at least $1-\frac{1}{f}$. However, as we are going to show, Phase 2 does not guarantee that $u = 2^w$ with high probability.

In order to satisfy $u = 2^w$ at the end of Phase 2, the status of channel in every call in Phase 2 must be COLLISION. In other words, `Broadcast(`$2^w - \frac{2^w}{2}$`)`, `Broadcast(`$2^w - \frac{2^w}{4}$`)`, `Broadcast(`$2^w - \frac{2^w}{8}$`)`, ..., `Broadcast(`$2^w - 1$`)` should be executed. Let i be any integer satisfying $0 \le i \le w$. The status of the channel in the call `Broadcast(`$2^w - 2^i$`)` is NULL with probability at least

$$\left(1-\frac{1}{2^{2^w-2^i}}\right)^n > \frac{1}{e}\left(1-\frac{2^{2^i}}{n}\right)^{n-1} > e^{-2^{2^i}-1}.$$

Once `Broadcast(`$2^w - 2^i$`)` is NULL then u does not exceed $2^w - 2^i$ at the end of Phase 2. In other words, with probability at least $e^{-2^{2^i}-1}$, the inequality $u \le 2^w - 2^i$ holds after the first phase. Writing $e^{-2^{2^i}-1} = \frac{1}{f}$, we have $2^i = \log(\ln f - 1)$, and $2^u \le 2^{2^w-2^i} = \frac{n}{\ln f - 1}$. Note that $f = e^{2^{2^i}+1}$ and $0 \le i \le w$, and thus $e^3 \le f \le e^{n+1}$. If the status of the channel in the call `Broadcast(`$2^w - \log(\ln f - 1)$`)` is NULL, then a call to `Broadcast(`u`)` in Phase 3 elects a leader with probability at most

$$\begin{aligned}
&\binom{n}{1}\left(1-\frac{1}{2^u}\right)^{n-1}\frac{1}{2^u} \\
\le\ & \left(1-\frac{\ln f - 1}{n}\right)^{n-1}(\ln f - 1) \\
<\ & 2\left(1-\frac{\ln f - 1}{n}\right)^{n}(\ln f - 1) \\
<\ & 2(\ln f - 1)e^{-(\ln f - 1)} < 2e\frac{\ln f - 1}{f}
\end{aligned}$$

Thus, t calls fails to elect a leader with probability at least

$$\begin{aligned}
&\left(1-2e\frac{\ln f - 1}{f}\right)^t \\
>\ & \left(1-2e\frac{\ln f - 1}{f}\right)\left(1-2e\frac{\ln f - 1}{f}\right)^{t-1} \\
>\ & (1-\frac{4}{e^2})e^{-2et\frac{\ln f - 1}{f}} > e^{-2et\frac{\ln f - 1}{f}-1}
\end{aligned}$$

Let $\frac{1}{f} = e^{-2et\frac{\ln f - 1}{f}-1}$. Then, t must be at least $\Omega(f)$. Hence, with probability at least $\frac{1}{f}$, Phase 3 runs in at least $\Omega(f)$ time slots. Recall that in Phase 2 the status of the channel in the call `Broadcast(`$2^w - \log(\ln f - 1)$`)` is NULL with probability $\frac{1}{f}$. Further, if such is the case, Phase 3 runs for at least f time slots with probability at least $1-\frac{1}{f}$. Consequently, Willard's protocol runs in at least $\Omega(f)$ time slots with probability at least $1-\frac{1}{f^2}$ for every $e^3 \le f^2 \le e^{n+1}$. Therefore, we have the following result.

Lemma 4.1 *Willard's uniform leader protocol runs in at least $\Omega(\sqrt{f})$ time slots with probability at least $1-\frac{1}{f}$ for every $f \le e^{\frac{n+1}{2}}$.*

5 Simulation results

The main goal of this section is to offer yet another perspective of the relative performance of Willard's and our uniform leader election protocols. Both protocols were simulated and the results are captured in Tables 1 and 2 in the form of a histogram. Each protocol was run 1,000,000 times for each of the values

$$n = 2, 10, 100, 1000, 10000, 100000, \text{and } 1000000.$$

It is easy to see that the two uniform leader election protocols have almost the same performance in terms of the expected number of time slots. However, the simulation results show that Willard's protocol is extremely slow with some probability as we have proved mathematically. In this simulation among the 1,000,000 runs, the largest number of time slots taken by Willard's protocol exceeds 1,600, while our leader election never takes more than 52 time slots.

References

[1] R. Bar-Yehuda, O. Goldreich, and A. Itai, Efficient emulation of single-hop radio network with collision

Table 1. The performance of Willard's uniform leader election protocol

n =	2	10	100	1000	10000	100000	1000000
1	500311	9671	0	0	0	0	0
2	218611	185948	0	0	0	0	0
3	117264	272504	10498	0	0	0	0
4	66842	159747	262230	78310	0	0	0
5	40275	96389	90190	13632	130218	332396	8
6	23098	93260	206693	246269	0	6	248
7	13257	56369	116681	57679	310615	160351	3625
8	8051	40112	96266	138792	76010	31610	28829
9	4833	26302	63966	121008	128376	62030	285704
10	2825	17789	44258	99221	113483	82863	178228
11	1692	12082	30465	68730	76535	100160	146388
12	1132	7922	21013	47703	51356	71373	107042
13	669	5641	14823	33794	34769	48303	72967
14	426	3980	10423	23684	23081	32846	50350
15	233	2883	7464	17030	15689	22603	35477
16	155	2037	5371	12164	10970	15419	24585
17	119	1429	4048	8800	7651	10845	17614
18	50	1127	3033	6718	5434	7671	12292
19	61	796	2178	4861	3733	5349	8892
20	32	654	1739	3721	2692	3974	6545
21	15	518	1410	2827	1989	2866	4732
22	16	390	1049	2198	1479	2064	3511
23	8	351	888	1768	1091	1551	2647
24	6	267	658	1471	767	1136	2002
25	2	205	598	1197	617	875	1607
26	5	165	449	936	463	669	1202
27	4	140	385	826	396	531	895
28	0	106	310	699	328	406	734
29	1	83	273	547	266	308	517
30	1	79	241	490	212	283	448
31	1	72	181	487	179	217	339
32	1	51	154	349	149	177	291
33	1	52	161	328	134	157	227
34	0	33	109	301	113	129	193
35	1	40	102	257	89	98	151
36	0	32	87	255	81	107	131
37	0	16	79	219	73	77	118
38	0	32	70	196	68	68	95
39	0	21	61	196	51	61	73
40	0	13	48	153	42	55	76
41- 50	1	105	331	1191	319	251	365
51- 60	1	71	146	495	184	53	143
61- 70	0	32	104	240	90	24	93
71- 80	0	49	72	123	66	9	55
81- 90	0	35	72	45	41	6	41
91-100	0	33	67	26	29	0	42
101-200	0	236	367	35	71	20	206
201-300	0	76	124	11	1	3	100
301-400	0	41	41	7	0	0	61
401-500	0	6	16	2	0	0	29
501-600	0	4	6	1	0	0	29
601-700	0	3	0	3	0	0	18
701-800	0	0	1	1	0	0	15
801-900	0	1	0	1	0	0	5
901-1000	0	0	0	1	0	0	8
1001-1100	0	0	0	1	0	0	1
1101-1200	0	0	0	1	0	0	2
1201-1300	0	0	1	0	0	0	1
1401-1500	0	0	0	0	0	0	1
1501-1600	0	0	0	0	0	0	1
1601-1700	0	0	0	0	0	0	1
AVE	2.185	4.677	7.043	8.975	8.984	8.870	11.582

Table 2. The performance of our uniform leader election protocol

n =	2	10	100	1000	10000	100000	1000000
1	500306	9767	0	0	0	0	0
2	218693	185832	0	0	0	0	0
3	77879	272252	10500	0	0	0	0
4	94007	166660	261358	78236	0	0	0
5	24528	99181	90097	13523	130451	331851	4
6	40459	94877	207357	246054	0	14	237
7	9803	49330	116821	57902	310312	160656	3784
8	16231	41589	102912	142546	75895	31535	28996
9	4125	23664	63662	124939	128710	62445	286019
10	6643	19146	44339	100638	114721	77610	178482
11	1655	10970	31486	72625	72921	102026	145891
12	2665	9019	21803	48872	52176	71009	106619
13	680	5239	15208	35224	34301	50991	76610
14	1103	4310	10569	23811	24930	33729	53468
15	273	2466	7372	17211	16699	23946	36242
16	460	1966	4988	11552	12190	16323	25502
17	97	1112	3467	8325	7986	11605	17540
18	196	860	2454	5474	5811	8026	12303
19	38	521	1773	3988	3870	5600	8549
20	88	401	1195	2787	2785	3822	6153
21	13	251	832	1904	1809	2663	4155
22	29	199	538	1262	1352	1815	2879
23	6	121	390	939	951	1337	1998
24	11	85	286	631	663	918	1363
25	2	50	163	446	426	636	968
26	2	43	133	306	322	445	693
27	1	36	95	262	222	327	461
28	4	21	59	165	129	212	331
29	1	10	29	102	116	150	236
30	1	7	30	85	70	96	156
31	0	5	21	69	51	61	99
32	1	6	23	49	33	47	67
33	0	2	7	23	28	30	56
34	0	0	6	19	27	21	45
35	0	0	7	10	11	12	23
36	0	2	4	6	12	14	17
37	0	0	2	5	4	6	15
38	0	0	4	1	4	8	14
39	0	0	2	2	3	5	5
40	0	0	4	3	3	2	6
41	0	0	0	2	1	0	3
42	0	0	0	0	0	0	3
43	0	0	0	0	1	1	2
44	0	0	0	1	0	1	3
45	0	0	2	1	2	0	1
46	0	0	2	0	0	0	0
47	0	0	0	0	1	0	1
48	0	0	0	0	0	1	1
49	0	0	0	0	0	1	0
50	0	0	0	0	0	2	0
52	0	0	0	0	1	1	0
AVE	2.310	4.510	6.795	8.672	8.921	8.834	11.321

detection on multi-hop radio network with no collision detection, *Distributed Computing*, 5, (1991), 67–71.

[2] J. Bentley and A. Yao, An almost optimal algorithm for unbounded search, *Information Processing Letters*, 5, (1976), 82–87.

[3] D. Bertzekas and R. Gallager, *Data Networks*, Second Edition, Prentice-Hall, 1992.

[4] R. M. Metcalfe and D. R. Boggs, Ethernet: distributed packet switching for local computer networks, *Communications of the ACM*, 19, (1976), 395–404.

[5] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.

[6] K. Nakano and S. Olariu, Randomized $O(\log\log n)$-round leader election protocols in radio networks, *Proc. of International Symposium on Algorithms and Computation*, (LNCS 1533), 209–218, 1998.

[7] K. Nakano and S. Olariu, Randomized leader election protocols for ad-hoc networks, *Proc. of Sirocco 7*, June 2000, 253–267.

[8] K. Nakano and S. Olariu, Randomized leader election protocols in radio networks with no collision detection, *Proc. of International Symposium on Algorithms and Computation*, 362–373, 2000.

[9] D. E. Willard, Log-logarithmic selection resolution protocols in a multiple access channel, *SIAM Journal on Computing*, 15, (1986), 468–477.