Leader Election in Anonymous Radio Networks: Model Checking Energy Consumption

Haidi Yue* and Joost-Pieter Katoen

Software Modeling & Verification Group
RWTH Aachen University
D-52056 Aachen, Germany
Phone:+49 421 80 21202
Fax: +49 241 80 222 17
{haidi.yue|katoen}@cs.rwth-aachen.de

Abstract. Leader election has been studied intensively in recent years. In this paper, we present an analysis of a randomized leader election using probabilistic model checking with PRISM. We first investigate the quantitative properties of the original protocol such as the expected number of election rounds. Then we modify the protocol so that it consumes less energy and process with larger energy amount has higher probability to be elected. The modified protocol is modeled as Markov Decision Process, which allow us to compute minimum and maximum values, interpreting the best- and worst-case performance of the protocol under any scenarios.

Keywords: leader election, power/performance modeling, stochastic models, statistical Analysis, radio networks

1 Introduction

Leader election is a fundamental problem in distributed computing, it was first proposed by Le Lann [12], who also gave the first solution. The problem consists of designating a particular process as the "organizer" of some task distributed among a group of processes. It requires that all processes in the network have the same local algorithm. The processes communicate through message exchange and at the end of the computation, the algorithm reaches a terminal configuration with exactly one process in a special state "leader", while all other processes are in the non-leader state. The process in the leader state is called the leader and other processes are aware of who gets the leadership.

A large range of leader election protocols exists. They can be either asynchronous or synchronous, anonymous or with unique identities, and the network topology can either be ring, tree, or complete graph. The complexity of these protocols can be measured either by the number of messages exchanged or by the

* The author is currently visiting professor Yusheng Ji's Lab in National Institute of Informatics, Tokyo, 101-8430 Japan.

time necessary to elect a leader. For more information we refer to [21]. From the point of view of energy efficiency, energy consumption can also be a criterion for protocol complexity, especially for wireless networks where energy consumption is an important issue. In this paper, we study the randomized leader election protocol [2] for synchronous anonymous radio networks, where each process is equipped with a single transceiver, and the network topology is not specified since all stations communicate via a unique channel.

We focus our study on energy consumption and use probabilistic model checking for the analysis. Probabilistic model checking is a formal verification technique for the modeling and analysis of systems with stochastic behavior. It has been widely applied in the design and analysis of randomized algorithms, communication and security protocols, biological systems and many others. In contract to simulation based approaches, probabilistic model checking searches exhaustively the whole state space of the systems, provides an exact, rather than approximated quantitative results. As an inevitable pay-off, probabilistic model checking has the limitation of system complexity. The frequently used probabilistic model checkers include PRISM [25], MRMC [24, 11] and VESTA [26]. In this paper, we model the protocol in terms of reactive modules and analyze it using PRISM.

PRISM is a probabilistic model checker developed at the University of Birmingham and Oxford. It provides support for three types of probabilistic models: DTMCs, CTMCs and MDPs, plus extensions of these models with costs and rewards. Models are described in the PRISM modeling languages, a simple, state based language, and properties are specified in a logic incorporates LTL, PCTL, CSL, etc. PRISM comprises the symbolic data structures and algorithms, based on *Binary Decision Diagrams*(BDDs) [4] and *Multi-Terminal Binary Decision Diagrams* (MTBDDs) [5]. These allow compact representation and efficient manipulation of large, structured model. For example, in [6], systems with over 10^{30} states have been verified. Further more, PRISM also features a discrete-event simulation engine, generating approximate results through Monte Carlo methods and sampling.

This paper starts with a description and analysis of the protocol in [2], then considers its energy consumptions, and finally proposes two amendments to optimize its energy aspects. In summary, the main contributions of this paper are: (1) We consider different channel failure scenarios and calculate the failure probability for unreliable channels; (2) Since energy usage is a big issue in the field of wireless networks, we improved the original protocol so that it consumes much less energy to elect leader than before. (3) We further propose an adaption of the protocol to increase the likelihood of elect a leader with maximal remaining energy.

The paper is organized as follows. We first introduce the original protocol and analyze it in the Section 2. A generalized version of the original protocol is discussed in Section 3. In Section 4, we introduce and discuss our modification of the original protocol. We conclude with a discussion in Section 6.

Related work. model checking and PRISM has been used in [7, 8] to verify and simply leader election protocol for anonymous ring networks. The HAVi Leader Election Protocol is modeled and analyzed in [18]. PRISM has also been used to verify a wide range of different wireless protocols, for example the CSMA/CD protocol [13] and a gossip-based protocol [14]. See the PRISM publication repository [25] for more detailed and future examples. Other application of formal methods for leader election protocol can be found in [17, 3].

To our knowledge, our paper is the first one about model checking leader election protocol for wireless networks with unknown network size, rather than for a fixed topology such as a ring. Furthermore, we evaluate the protocol by focusing on the energy consumption until a leader has been elected.

2 Leader election for fixed network size

In this section, we first introduce the randomized leader election protocol of [2], then model and analyze it with PRISM.

2.1 Protocol introduction

We consider the randomized leader election protocol introduced in [2] which is designed for radio networks, in which every station is equipped with only one transceiver, so that a station cannot perform transmitting and listening operations at the same time. This means that a station can not detect collisions while transmitting. The assumptions made by this protocol are:

1. The stations are identical and cannot be distinguished by serial or manufacturing number.
2. Time is slotted and all stations have a local clock that are synchronized.
3. The network has no collision detection capabilities.
4. The single channel which is available to all stations is reliable. This means that messages are not lost, cannot be reordered, and are not duplicated.

We say the status of the channel is SINGLE if only one station is transmitting in the current time slot. Otherwise, if no station is transmitting or more than one stations are transmitting in the same time slot, the channel is recognized as NOISE. We denote by S the set of all stations and assume $|S| = N \geqslant 2$. Two scenarios are distinguished: (1) The network size N is known by all stations or (2) an upper bound of N is known by all stations.

We first consider the case when the number N of stations in the network is known in advance. Protocol *Leader-Election(N)* elects a leader with known N and consists of two phases: a partition phase and an election phase. In the partition phase, the set of stations will be randomly partitioned into 3 disjoint sets A, B and S-A-B. A leader will be elected in the election phase if the partition formed in the partition phase satisfies $|A| = |B| = 1$. Otherwise, a new election round will be initiated. The detailed election scheme is outlined below.

<u>*Leader-Election(N)*</u>:

Partition phase :

- **step** 1 : Every station tosses a fair coin and belongs to A with probability $\frac{1}{N}$.
- **step** 2 : Every station that is not in A tosses a fair coin and belongs to B with probability $\frac{1}{N-1}$.
- **step** 3 : Stations that belongs neither to A nor to B after step 2 are in S-A.

Election phase :

- **slot** 1 : Every station in A broadcasts on the channel. Stations in S-A monitor the channel.
- **slot** 2 : If the channel was SINGLE in slot 1, every station in B broadcasts on the channel and processes in S-B monitor the channel.
- **slot** 3 : If the channel was SINGLE in slot 2, every station in A broadcasts on the channel and announces itself as the leader, stations in S-A-B monitor the channel and get informed that a leader is elected.

We model the above protocol as a *discrete-time Markov Chain* (DTMC) and analyze it using the probabilistic model checker PRISM. A DTMC is a transition system which labels each transition with a probability such that the sum over all the outgoing transitions for each state equals one. The behavior of DTMC is fully probabilistic. Hence, we can define a probability space over infinite paths through the model and quantitative analyze the likelihood that a particular event occurring. For a more detailed discussion we refer to [1].

In the setting of probabilistic model checking, the properties of a system are typically expressed in temporal logic, such as PCTL [9]. For example, the following formula:

$$\mathbf{P} =? \, [\mathbf{F}^{\leqslant k} \text{ ``}error\text{''}]$$

represents the probability that an error state is reached within k steps. When the states or transitions are labeled with some rewards or costs, TDMCs can also be used to reason about a wide range of quantitative measures, such as "passed election rounds", or "energy consumed". For example:

$$\mathbf{R}\{\text{``rounds''}\} =? \, [\mathbf{F} \text{ ``}terminated\text{''}]$$

represents the expected number of rounds until termination.

In the following, we represent the model checking results of the above protocol.

2.2 Scalability

If the channel is reliable and the network size N is known exactly, *Leader-Election(N)* provides a quite good scalability. Given a fixed $N > 1$, the probability p_N of $|A|$=$|B|$=1 after partition can be calculated by [2]:

$$\begin{aligned} p_N &= \binom{N}{1}\frac{1}{N}(1-\frac{1}{N})^{N-1}\binom{N-1}{1}\frac{1}{N-1}(1-\frac{1}{N-1})^{N-2} \\ &= (1-\frac{1}{N})^{N-1}(1-\frac{1}{N-1})^{N-2} \\ &> \frac{1}{e^2} \end{aligned} \tag{1}$$

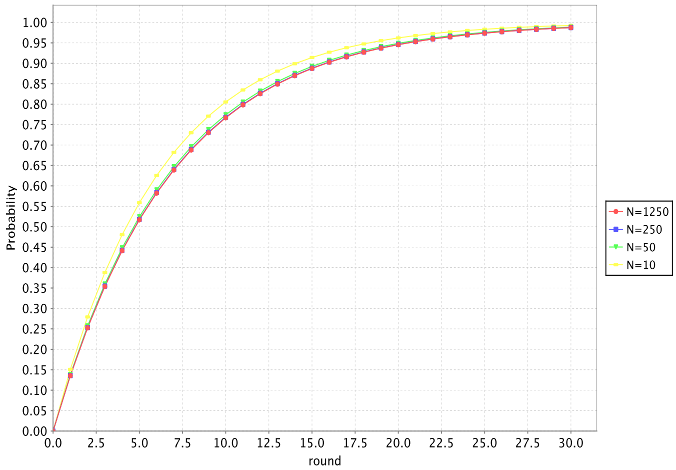

Fig. 1. good scalability

Since the election phase is deterministic, to calculate the probability of successful election, it is sufficient to model the partition phase as a simple DTMC with only two states: from the initial state, with probability p_N, it goes to the second state(i.e. an unique leader is elected), otherwise, with probability $1-p_N$, it stays in the initial state. Figure 1 shows the probability of successful election of a leader (the y-axis) within r rounds (the x-axis), for different number N of stations. As we can see, even for network with large N (i.e. $N = 1250$), the probability of successful election converges almost as fast as for network with small size. This shows that the protocol has a good scalability.

Suppose the first r executions of *Leader-Election(N)* failed to elect a leader. Since each attempt is independent, the probability of this occurring is at most $(1-p_N)^r < e^{-r \cdot p_N}$. It follows that with probability exceeding $1-e^{-r \cdot p_N}$, the protocol elects a leader within at most r rounds. For f satisfying $f = e^{-r \cdot p_N}$, the protocol terminates, with probability exceeding $1-f$ in $-\frac{1}{p_N}\ln f$ rounds.

Lemma 1. *Let X be a discrete random variable taking a value at most $T(F)$ with probability at most F, where T is a non-decreasing function, then, $E(X) \leqslant \int_0^1 T(F)dF$.*

Thus, the expected number of election rounds to terminate is bounded by

$$\int_0^1 -\frac{1}{p_N} \ln f \, df = \frac{1}{p_N} < e^2 < 8,$$

which means that with increasing N, the protocol can on average elect a leader within 8 election rounds.

2.3 Unreliable channel

The protocol [2] assumes that the communication channel is reliable, however, in the real world, this is mostly not the case. For instance, during the sending of one station, some background noise may screen out the channel. As a consequence, although the channel is supposed to be SINGLE, other stations still evaluate it as NOISE. Or, it can also happen that more than one station is sending but due to unexpected weather, the signal power attenuates immensely, and only one station can access the channel and the channel becomes SINGLE.

Following the above considerations, we introduce two kinds of channel failures:

1. SINGLE to NOISE : There is only one station sending to the channel, however other stations that are monitoring consider the channel to be NOISE.
2. NOISE to SINGLE : There are at least two stations broadcasting to the channel, but other stations which are monitoring the channel receive information from only one station that is sending and consider the channel to be SINGLE.

In fact, the above two scenarios are the same in the sense that we define channel failure as: the number of stations that successful broadcasting to the channel is smaller than the number of stations that attempt to broadcast to the channel. I.e., if the channel is NOISE due to nobody is sending at that moment, it can not happen that other stations observe the channel as SINGLE.

Assume that per election round, a channel failure occurs at most once, either in slot 1, slot 2 or slot 3. We model this in PRISM by introducing a rate p_c indicating that the channel works correctly with probability p_c. When $p_c = 1$, the channel is reliable. If $p_c < 1$, a channel failure can occur with probability $1-p_c$ in slot i ($i \in \{1, 2, 3\}$) either be the reason of SINGLE to NOISE or NOISE to SINGLE.

When a channel failure occurs, although the protocol terminates on average with almost the same number of election rounds, it could terminate incorrectly, i.e., more than one leader is elected. Let π be the probability of correct termination, Table 1 shows the model checking results for π with $p_c = 0.95$ and $N = 10$, under different failure types and different slots.

If follows from Table 1 that when SINGLE to NOISE occurs, if it occurs in the first or the second slot, the protocol can still terminate correctly. However, if it occurs in the third slot, with some probability, more than one leader is elected. Indeed, this happens when $|A| = |B| = 1$, then in the third slot, since the channel was SINGLE in the last two slots, the unique station in A understands itself as leader and announces this to other stations. If now SINGLE to NOISE failure arises, other stations, especially the single station in B, will consider $|B| \neq 1$ and start a new election round and eventually elect another station as leader. This kind of event happens in each round with probability $p_N(1 - p_c)$.

failure type \ slots	1	2	3
SINGLE to NOISE	1.0	1.0	0.9547
NOISE to SINGLE	0.9672	1.0	1.0

Table 1. correctness result with unreliable channel

Another scenario of incorrect termination appears when failure NOISE to SINGLE in the first slot occurs. This occurs when $|A| > 1$ and $|B| = 1$. A NOISE to SINGLE failure in the first slot results in a wrong decision of the unique station in B, it hence broadcasts to the channel that slot 1 was SINGLE. Now all stations in A consider themselves as leader, whereas all other stations receive nothing in the third slot and start a new election round.

3 Leader election for networks of unknown size

The protocol introduced above is considered for networks with known total number of stations N, [2] also discusses the case if only the upper bound u of N is available. In this case, the following protocol *Leader-Election(*$2^1, 2^2, ..., 2^{\lceil logu \rceil}$*)* will be executed, which is a generalization of *Leader-Election(N)*.

Leader-Election($2^1, 2^2, ..., 2^{\lceil logu \rceil}$*)*
for $i = 2^1$ **to** $2^{\lceil logu \rceil}$ **do**
  run *Leader-Election(i)*
  terminate **if** leader is elected
od

[2] does not mention what happens if no leader is elected after *Leader-Election(*$2^{\lceil logu \rceil}$*)* is performed. Although this occurs with a very low probability, we need to specify in our model which action should be undertaken if it is encountered. Hence we assume that once *Leader-Election(*$2^{\lceil logu \rceil}$*)* is executed and no leader is elected yet, i will be set back to 2 and the algorithm *Leader-Election(*$2^1, 2^2, ..., 2^{\lceil logu \rceil}$*)* starts again.

Let $n = \lceil log\ u \rceil$ be the smallest number exceeding u which is a power of 2. Figure 2 shows the probability of elect a leader (y-axis) at each round (x-axis) in a network with $N = 12$ stations.

The red curve indicates the case when the network size is known. The blue one presents the case if we assume $n = 16$, and the green one shows the case

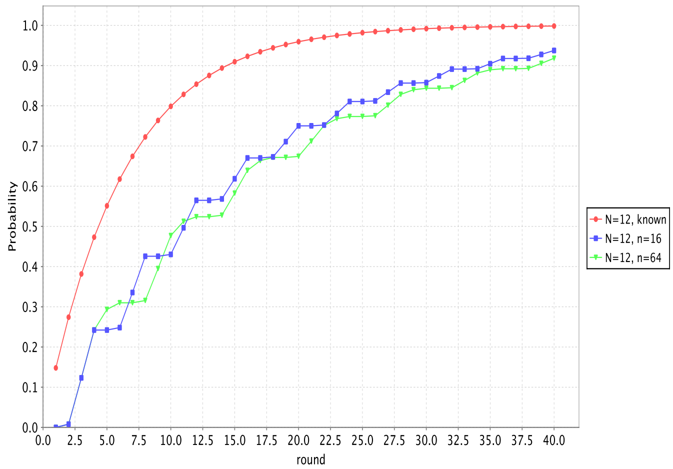

Fig. 2. Impact of knowledge of n.

when $n = 64$. As we can see, once we do not know the exact number N, the probability of elect a leader at the same round descends significantly, and it takes longer to elect a leader, since the blue and green curves converges slower than the red one. Because the energy consumption per election round is the same, it follows that for networks with unknown network size, leader election consumes in general more energy to elect leader. However, the difference between the two bounds $n = 16$ and $n = 64$ is not huge, i.e., if we do not know the exact number of N, a coarse estimation performs almost the same as a more accurate one.

Let $p(n, N)$ be the probability of $|A| = |B| = 1$ after the partitioning phase, with $N > 1$ the exactly number of stations and $n > 1$ the number used to form the partition.

$$p(n, N) = \binom{N}{1} \cdot \frac{1}{n} \cdot (1 - \frac{1}{n})^{N-1} \cdot \binom{N-1}{1} \cdot \frac{1}{n-1} \cdot (1 - \frac{1}{n-1})^{N-2}$$

Obviously, when $n = N$, $p(n, N)$ is equal to equation (1). Figure 3 shows the value of $p(n, N)$ (y-axis) with $n = 5, 15, 80$ and $N \in \{2, ..100\}$. For fixed N, $p(n, N)$ has its maximal value if $N = n$, and for $N = n = 2$, $p(n, N)$ has the maximal value $\frac{1}{2}$ (not shown in figure). This means that if we do not know the exact number N of stations in the network, and use another number $n \neq N$ to form the partition, then the probability of $|A| = |B| = 1$ can be much smaller than $p(n, N)$, and more election rounds are needed. In the worst case, if $N > 2$ and we start leader election by executing Leader-Election(2^1),..., then for sure no leader will be elected in the first election round, because $p(2, N) = 0$.

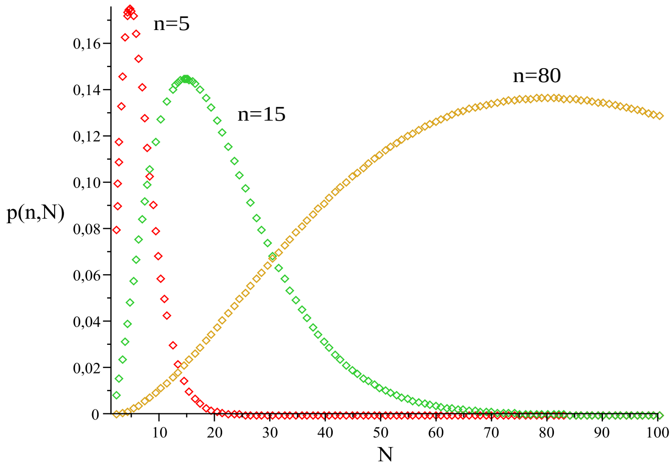

Fig. 3. Probability of $|A| = |B| = 1$ with different n and N.

4 Energy consumption

4.1 Energy inefficiency of the protocol

Each station has two activities that take energy: transmitting or monitoring. From the point of view of energy consumption, the protocol introduced above is not really energy efficient, in the sense that it exists superfluous monitoring actions that consume energy. Consider Table 2: in all time slots, all stations are either monitoring or transmitting. However, the monitoring action of stations in S-A-B in the first and second slot does not contribute to the election procedure at all. Without these actions, the probability of successfully electing a leader will not be changed.

Slot \ Partition	A	B	S-A-B
slot 1	send	monitor	monitor
slot 2	monitor	send	monitor
slot 3	send	monitor	monitor

Table 2. action table

4.2 Energy improvement

It follows that in the election phase, it is not necessary for stations in S-A-B to monitor the channel all the time. This suggests to modify the protocol by letting the stations in S-A-B idle during the first two slots and only monitor in the third slot to eventually receive the information from A. This neither affects the correctness of the protocol nor changes the probability of successful election in each round.

To model energy consumption, we assume a send action consumes J units of energy, a monitor action consumes $\alpha_1 J$, an idle slot consumes $\alpha_2 J$ and the switching on and off of a transceiver costs $\alpha_3 J$ energy units. Usually, especially for sensor networks, the factor α_1 ranges from $1.0 - 1.5$ and factor α_2 is a thousand times smaller than 1 [20, 16, 19].

In the sequel, if not stated otherwise, we consider leader election in network with known size N. We first analyze the average energy consumption in the original protocol [2]. The energy consumption of each station in each slot is given in Table 3. Let $X_{i,j}$ be a random variable denoting the energy consumption of station i in round j. Obviously, $E[X_{i,j}]$ is equal to $E[X_{k,l}]$ for $k \neq i$ or $l \neq j$ and it holds:

Slot \ Partition	A	B	S-A-B
slot 1	J	$\alpha_1 J$	$\alpha_1 J$
slot 2	$\alpha_1 J$	J	$\alpha_1 J$
slot 3	J	$\alpha_1 J$	$\alpha_1 J$

Table 3. energy consumption of the original protocol

$$
\begin{aligned}
E[X_{i,j}] &= \left(\frac{1}{N}(2J + \alpha_1 J) + \frac{1}{N}(J + 2\alpha_1 J) + \frac{N-2}{N} 3\alpha_1 J \right) \\
&= J \left(\frac{2}{N} + \frac{1}{N}\alpha_1 + \frac{1}{N} + \frac{2}{N}\alpha_1 + (3 - \frac{6}{N})\alpha_1 \right) \\
&= J \left(\frac{3}{N} + (3 - \frac{3}{N})\alpha_1 \right)
\end{aligned}
$$

and $\lim_{N \to \infty} E[X_{i,j}] = 3\alpha_1 J$.

Slot \ Partition	A	B	S-A-B
slot 1	J	$\alpha_1 J$	$\alpha_2 J + \alpha_3 J$ (switch off)
slot 2	$\alpha_1 J$	J	$\alpha_2 J + \alpha_3 J$ (switch on)
slot 3	J	$\alpha_1 J$	$\alpha_1 J$

Table 4. energy consumption of the modified protocol

Now consider the energy Table 4 of the modified protocol with idle periods. For this protocol, the expected energy consumption $E'[X_{i,j}]$ of station i in round j is:

$$
\begin{aligned}
E'[X_{i,j}] &= \left(\frac{1}{N}(2J + \alpha_1 J) + \frac{1}{N}(J + 2\alpha_1 J) + \frac{N-2}{N}(2\alpha_2 J + 2\alpha_3 J + \alpha_1 J) \right) \\
&= J \left((\frac{3}{N} + \frac{3}{N}\alpha_1) + \frac{N-2}{N}(\alpha_1 + 2\alpha_2 + 2\alpha_3) \right)
\end{aligned}
$$

and $\lim_{N \to \infty} E'[X_{i,j}] = (\alpha_1 + 2\alpha_2 + 2\alpha_3)J$.

To get an improvement towards energy efficiency, i.e.,

$$\lim_{N\to\infty} E'[X_{i,j}] < \lim_{N\to\infty} E[X_{i,j}]$$

it should hold:

$$\alpha_2 + \alpha_3 < \alpha_1$$

which is usually the case [20, 16, 19].

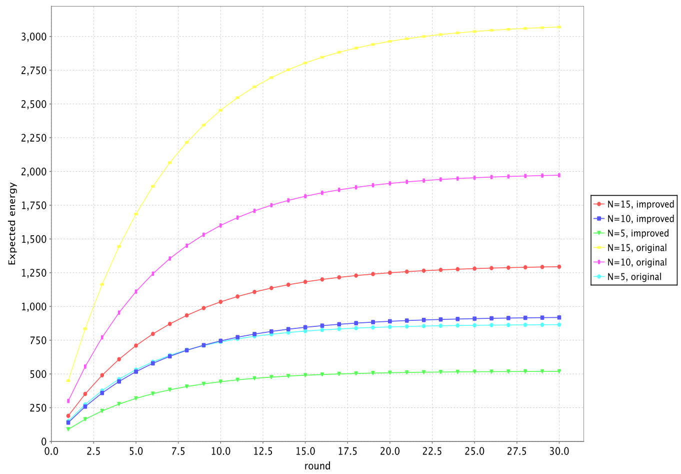

Fig. 4. Standby in *S*-*A*-*B*

Assume $J = 10$ energy units. Figure 4 shows the expected energy consumption (y-axis) at each election round (x-axis) until successfully a leader has been elected, with network size $N = 5$, 10 or 15. Curve labeled with "original" indicate the result of the original protocol and curves labeled with "improved" presents the modified protocol with $\alpha_1 = 1$, $\alpha_2 = \alpha_3 = 0$, which are quite ideal factors, but still, the energy difference is so huge and the factor α_2, α_3 in real application is small enough to confirm that our modification is more energy efficient than the original one. In general, if consider $\alpha_1 = 1$, $\alpha_2 = \alpha_3 = 0$, it holds that $\lim_{N\to\infty} \frac{E'[X_{i,j}]}{E[X_{i,j}]} = \frac{1}{3}$, which means that the modified protocol consumes only one third energy as the original one.

4.3 Elect leader with higher energy

Besides introducing idle moments for station in *S*-*A*-*B* in the protocol, it also makes sense if the algorithm tries to elect a leader with maximal energy level, since the role leader is usually expected to perform some special tasks which consumes extra energy. In the following, we modify the original protocol by partitioning the stations into different energy levels and let stations in higher energy level have a higher chance to be elected as leader.

Let M be the maximal possible energy storage available in the current network and assume there are b energy levels. The lowest energy level is b and the highest energy level is 1. A station s belongs to energy level l, if $\frac{M}{b}(b-l) < s_e \leqslant \frac{M}{b}(b-l+1)$, where $s_e > 0$ is the energy status of s. The underlying assumptions of this modification are:

– Each station has the knowledge of M. This can be realized by for example a message from the base station.
– Each station knows its energy level and this level will not change during the election process. This is also possible because each energy level covers a range of energy status. Even when considering battery recharge during idle slots [22, 10], the possibility that stations change their energy level is low.
– There are at least two stations in the energy level 1. This assumption is also reasonable, because this is usually the energy level of the leader from the last leader election call, and a successful leader election requires at least two stations to participate.

The modified leader election protocol *Leader-Election-High-Energy(N)* works as follows:

Station s calculates and belongs to energy level $l_s \in \{1, ..., b\}$ with respect to its current energy status.
for $i = 1$ **to** b **do**
  if $l_s \geqslant i$
    execute *Leader-Election*($\frac{N}{b}i$)
    terminate **if** a leader is elected
  else only wake up in the third slot.
od
If no leader is elected after b rounds, all stations starts *Leader-Election(N)*.

Intuitively speaking, a station will participate in the election if its energy level is greater than the round number, otherwise, it plays the role of stations in *S-A-B*. Since we do not know how the energy level actually is distributed in the network, the protocol assumes that the energy level of stations are uniformly distributed, i.e., in election round i, it supposes that $\frac{N}{b}i$ stations take part in the election process. In the beginning, only stations belonging to higher energy level are allowed to participate in the election procedure. This increases the probability of being leader for stations with higher energy storage, because they participate in an election phase more often.

We model *Leader-Election-High-Energy(N)* by a Markov decision process (MDP), which is an extension of DTMC with the ability also representing nondeterministic behavior. The nondeterminism is necessary to obtain different energy level distributions. More precisely, at the beginning of an election, each station first selects nondeterministically its energy level, then starts random partition with given probability. Due to the presence of nondeterminism, for MDPs, we can not compute the probability unless the nondeterministic choice is resolved. Instead, the analysis of MDP models provide minimum and maximum probability that an event occurring. In our case, we compute the minimal and maximal expected number of election rounds to elect a leader, which represents the best and worst results respectively, under different energy level distributions.

b	max-rounds	min-rounds	max-prob.	min-prob.
1	6.42	6.42	0.25	0.25
2	7.07	6.21	0.34	0.25
4	8.18	4.08	0.69	0.25

Table 5. Model checking result for Leader-Election-High-Energy(N) with $N = 8$.

We model networks with $N = 8$, $b = 4$ and $N = 9$, $b = 3$, the state space of these two models are $613, 474, 725$ and $819, 009, 820$, respectively. For MDPs, besides the state space explosion problem, a more deciding parameter for verification feasibility is the number of nodes in MTBDD Matrix. The more nondeterministic choices there are, the larger the MTBDDs. For instance, to verify the reachability property, the model with $N = 8$, $b = 4$ has $206, 368$ MTBDD nodes whereas the model with $N = 9$, $b = 3$ has $141, 921$ MTBDD nodes. Hence it takes on average for $N = 8$, $b = 4$ more than 200 seconds to verify this property and for $N = 9$, $b = 3$ it takes less time. For networks with $N > 9$, $b > 4$, PRISM failed to build a model due to the lack of memory. The maximal and minimal number of expected election rounds for $N = 8$, $b = 1, 2$, or 4, as well as the maximal and minimal probability that an leader from the highest energy level is elected, can be found in Table 5. The columns “max-rounds”, “min-rounds” indicate the maximal and minimal number of expected election rounds to elect a leader, respectively. The two right most columns “max-prob.”, “min-prob.” indicate the maximal and minimal probability that the leader is from the highest energy level, respectively.

Obviously, when $b = 1$, the maximal and minimal values are the same. If $b \neq 1$, larger b yields a wider difference between maximum and minimum results. The worst case (which corresponds to the result with maximal number of expected election rounds) occurs when all stations belonging to the lowest energy level besides the two in the assumption. In this case, there are actually only two stations (the two we assume in the highest energy level) active in the first $b - 1$ election rounds, whereas the number used to form partition is unequal to 2, which reduces the probability of $|A| = |B| = 1$. The worst case (which corresponds to the result with minimal number of expected election rounds) occurs when in each block there are exactly $\frac{N}{b}$ stations. Then in each election round i, the number of stations that participate election is equal to the number used to form partition, hence the probability of $|A| = |B| = 1$ is maximal. The reason is that as mentioned before, the protocol assumes that the energy level is uniformly distributed, thus it has the best performance when the nondeterministic choice results in a uniform energy distribution.

5 Discussion and Conclusion

In this paper, we have presented the application of probabilistic model checking to a leader election protocol for wireless noisy radio networks with a single transceiver, focusing on the probability to elect a leader within a given number of rounds, and the expected energy consumption. All verification and experiments with PRISM were run on a 3.0GHz Pentium 4 processor with 2GB memory.

We improved the protocol by letting some stations sometimes idle. We have shown by both model checking and mathematical analysis that this improvement indeed reduce a large amount of energy consumption. Furthermore, we modified the protocol by partitioning the stations into different blocks with respect to their energy level, to increase the likelihood to elect a leader with higher remaining energy (e.g. battery capacity). We modeled the modified protocol as MDP and show that, if the energy status of stations are uniformly distributed, stations in the highest energy level have very high probability to be elected as leader, and the number of election rounds is less than the original protocol.

For future work, we plan to incorporate the battery models in [10] for stations in the protocol, and to model and verify a leader election protocol that can easily accommodate topology changes [15, 23].

References

1. Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. MIT Press, 2008.
2. Jacir Luiz Bordim, Yasuaki Ito, and Koji Nakano. Randomized leader election protocols in noisy radio networks with a single transceiver. In *Parallel and Distributed Processing and Applications(ISPA)*, volume 4330 of *LNCS*, pages 246–256. Springer-Verlag, 2006.
3. J. Brunekreef, J-P. Katoen, R. Koymans, and S. Mauw. Design and analysis of dynamic leader election protocols in broadcast networks. In *Distributed Computing 9(4)*, pages 157–171, 1996.
4. R. Bryant. Graph-based algorithms for boolean function manipulation. In *IEEE Transactions on Computers, C-35(8)*, pages 677–691, 1986.
5. E. Clarke, M. Fujita, P. McGeer, K. McMillan, J. Yang, and X. Zhao. Multi-terminal binary decision diagrams: An efficient data structure for matrix representation. In *International Workshop on Logic Synthesis (IWLS)*, pages 1–15, 1993.
6. L. de Alfaro, M. Kwiatkowska, G. Norman, D. Parker, and R. Segala. Symbolic model checking of concurrent probabilistic processes using MTBDDs and the kronecker representation. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 1785 of *LNCS*, pages 395–410. Springer, 2000.
7. W. Fokkink and J. Pang. Simplifying Itai-Rodeh leader election for anonymous rings. In *Automated Verification of Critical Systems (AVoCS)*, volume 128, pages 53–68, 2004.
8. W. Fokkink and J. Pang. Variations on Itai-Rodeh leader election for anonymous rings and their analysis in PRISM. In *Journal of Universal Computer Science*, volume 12, pages 981–1006, 2006.
9. H. Hansson and B.Jonsson. A logic for reasoning about time and reliability. In *Formal Aspect of Computing*, pages 512–535, 1994.

10. Marijn R. Jongerden, Boudewijn R. Haverkort, Henrik Bohnenkamp, , and Joost-Pieter Katoen. Maximizing system lifetime by battery scheduling. In *International Conference on Dependable Systems and Networks*, pages 63–77, 2009.
11. Joost-Pieter Katoen, Ivan S. Zapreev, Ernst Moritz Hahn, Holger Hermanns, and David N. Jansen. The ins and outs of the probabilistic model checker MRMC. In *Int. Conf. on the Quantitative Evaluation of Systems (QEST)*, pages 167–176, 2009.
12. Gerard Le Lann. Distributed systems: Towards a formal approach. In *Information Processing 77, Proc. of the IFIP Congress*, pages 155–160, 1977.
13. T. Herault R. Lassaigne F. Magniette S. Messika S. Peyronnet M. Duflot, L. Fribourg and C. Picaronny. Probabilistic model checking of the CSMA/CD protocol using PRISM and APMC. In *Automated Verification of Critical Systems (AVoCS), ENTCS*, volume 128, pages 195–214, 2004.
14. G. Norman M. Kwiatkowska and D. Parker. Analysis of a gossip protocol in PRISM. In *ACM SIGMETRICS Performance Evaluation Review*, volume 36, pages 17–22, 2008.
15. N. Malpani, Jennifer L. Welch, and N. Vaidya. Leader election algorithms for mobile ad hoc networks. In *Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 96–103. ACM, 2000.
16. Matthew J. Miller and Nitin H. Vaidya. Minimizing energy consumption in sensor networks using a wakeup radio. In *Wireless Communications and Networking Conference (WCNC)*, volume 4, pages 2335–2340, 2004.
17. J. Romijn. A timed verification of the IEEE 1394 leader election protocol. In *Formal Methods in System Design 19(2)*, pages 165–194, 2001.
18. J. M. T. Romijn. Model checking the HAVi leader election protocol, 1999.
19. Nordic Semiconductors. *nRF2401 Single-chip 2.4GHz Transceiver Data Sheet*, 2002.
20. Victor Shnayder, Mark Hempstead, Borrong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the power consumption of large-scale sensor network applications. In *Int. Conf. on Embedded Networked Sensor Systems*, pages 188–200. ACM, 2004.
21. Gerard Tel. *Introduction to Distributed Algorithms*. Cambridge University Press, 2000.
22. Doru E. Tiliute. Battery management in wireless sensor networks. In *Electronics and Electrical Engineering 4(76)*, pages 9–12, 2007.
23. S. Vasudevan, J. Kurose, and D. Towsley. Design and analysis of a leader election algorithm for mobile ad hoc networks. In *Int. Conf. on Network Protocols (ICNP)*, pages 350–360, 2004.
24. MRMC website. `http://www.mrmc-tool.org/trac/`.
25. PRISM website. `http://www.prismmodelchecker.org`.
26. VESTA website. `http://osl.cs.uiuc.edu/~ksen/vesta2/`.