

Leader Election Algorithms for Multi-channel Wireless Networks

Tarun Bansal, Neeraj Mittal, and S. Venkatesan

Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, USA
tarun@student.utdallas.edu, {neerajm,venky}@utdallas.edu

Abstract. We study the leader election problem in single-hop multi-channel wireless networks with single-antenna radio nodes. The objective is to elect leaders for all channels on which one or more nodes in the network can operate. We assume that nodes do not have collision detection capability. In this paper, we propose three algorithms for leader election: one deterministic and two randomized. The deterministic algorithm executes for at most $2N + M \lceil \log M \rceil$ time slots and guarantees leader election for all the channels, where M is the size of the universal channel set available to the nodes and N is the size of the label space used to assign unique labels to the nodes. The randomized algorithms guarantee that a leader is elected for all the channels with probability at least $1 - 1/f$ within $O(M \log^2(n_{\max}) + M \log^2(Mf))$ and $O(M \log(n_{\max}) \log(Mf))$ time slots, respectively, where n_{\max} is the maximum number of nodes operating on any channel in the network. To the best of our knowledge, this is the first work on leader election in single-antenna multi-channel radio networks.

Keywords: wireless networks, cognitive radios, multiple channels, leader election.

1 Introduction

Currently, the wireless channels are allocated using the fixed spectrum allocation policy in which the spectrum is assigned by a central authority to various services. However some frequencies are much more in demand than other owing to their lower energy requirements, low error rate and higher transmission range. As a result, a large portion of the spectrum is sporadically used whereas in some portions the usage is a lot more concentrated [1]. In order to allocate channels efficiently so as to avoid spectrum holes, the use of *cognitive radios* has been recommended [1,2].

Cognitive radios (CRs) are able to change their transmission and reception parameters to communicate efficiently avoiding interference with other users by sensing the spectrum. This adjustment can be done dynamically and can include changes in communication frequency, encoding, link layer parameters etc.

Even though cognitive radios are equipped with only a single antenna, still this dynamic adjustment allows them to operate on multiple channels.

While availability of multiple channels allow better network throughput potentially, but it also makes the setting up of the network much more difficult. This is because different nodes may have different capabilities. For example, a node R_1 may be able to communicate on channel C_5 , but a neighboring node R_2 may not be able to communicate on that channel because of different hardware capability or spatial variance of the channel availability set [1]. This makes it a lot more difficult to solve traditional problems for multi-channel radio networks. Leader election is one such fundamental problem. It involves selecting a *distinguished* node in the network. In wireless networks, leader election has been used to solve many other problems like routing [3], key distribution [4], coordination in sensor networks [5], neighbor discovery [6,7] and so on.

For multi-channel networks, a leader can be elected in two different ways: (i) one leader for all the channels, or (ii) one leader on each channel with possibly different leaders for different channels. Having one leader for all the channels is not feasible if no node in the network has the capability to communicate on all the channels. Therefore, to ensure that every node in the network can communicate with a leader of a channel in a single step, we have to ensure that there is a leader for every channel on which one or more nodes in the network can operate.

In this paper, we propose several leader election algorithms (one deterministic and two randomized) for single-hop multi-channel networks (like cognitive radio networks). The deterministic algorithm guarantees that a leader is elected for all channels using at most $2N + M \lceil \log M \rceil$ time slots, where M is the size of the universal channel set available to the nodes and N is the size of the label space used to assign unique labels to the nodes. Note that the time complexity of our deterministic algorithm is much lower than that of the naïve algorithm, which uses MN time slots. The randomized algorithms guarantee that a leader is elected for all the channels with probability at least $1 - \frac{1}{f}$ within $O(M \log^2(n_{\max}) + M \log^2(Mf))$ and $O(M \log(n_{\max}) \log(Mf))$ time slots, respectively, where n_{\max} is the maximum number of nodes operating on any channel in the network. Both randomized algorithms work without assuming the knowledge of n_{\max} or N . As it can be observed, the time complexity of both randomized algorithms is asymptotically much lower than that of the deterministic algorithm when $N \gg M \log^2(n_{\max}) + M \log^2(Mf)$.

The rest of the paper is organized as follows. In section 2, we discuss the previous work conducted on leader election in single channel networks. We describe our system model in section 3. In section 4 we describe a deterministic algorithm for leader election. In section 5, we describe two randomized algorithms for leader election. Finally we conclude the paper in section 6. Due to space constraints, some of the proofs have been omitted and can be found elsewhere [8].

2 Related Work

Depending on radio transceiver’s capability to detect collision, two radio models have been used in literature [9] for developing wireless algorithms viz. “Strong Radio model” and “Weak Radio Model”. In the weak radio model, nodes cannot distinguish between “no transmission” and “multiple transmissions”. This is also known as *Radio Network with no Collision Detection* (no-CD RN). Moreover, the transmitting nodes are not capable of monitoring the state of the channel simultaneously while transmitting. In strong radio model, nodes have collision detection capability which allows them to differentiate between “no transmission” and “multiple transmissions”. Strong radio model also allows transmitting nodes to detect collision by simultaneously allowing them to monitor the channel state. As pointed out by the authors [9], only the weak radio model is consistent with the IEEE 802.11 standards and current WLAN standards.

For deterministic leader election algorithms for single channel weak radio networks, Jurdzinski et al. [10] have proved that the lower bound on time requirements for such algorithms is $N - 1$ time slots. Nakano et al. [11] have given one such time optimal and energy efficient deterministic algorithm which terminates in optimal number of time slots.

A trivial way to design deterministic algorithm for leader election for multiple channels could be to run M instances of Nakano’s [11] algorithm on each channel. However, in that case the time complexity of the protocol would become $O(NM)$. In this paper, we present a deterministic algorithm for leader election which makes use of FDMA to achieve a much lower time complexity of $O(N + M \log M)$.

Different randomized algorithms for leader election have been proposed in the literature for different radio models. The leader election algorithms proposed in [10], [12], [13], [14] are based on strong radio model. They either assume that the nodes have collision detection capability or that the nodes can monitor the state of the channel at the same time while transmitting. In this paper, we consider only those algorithms that assume weak radio model.

Metcalfe et al. [15] have proposed algorithms for leader election which assumes the knowledge of the number of nodes present in the network. However for multi-channel networks, it is possible that the number of nodes present on different channels differ vastly. This may be due to different hardware capabilities of the devices or because of the spatial variance of the channel availability set [1]. For example, a higher percentage of nodes may be able to communicate on channels which have low energy requirements, low error rate etc. as compared to channels which do not exhibit these properties. This makes it very difficult to predict the number of nodes on each channel as required by their algorithm.

Hayashi et al. [16] have given a randomized algorithm for leader election for the weak radio model with unknown number of nodes present on the channel, which terminates in $O(\log^2 n)$ broadcast rounds with probability exceeding $1 - 1/n$ where n is the actual number of nodes present on the channel. Their algorithm proceeds in multiple rounds and each round is further divided into time slots. The number of time slots in a round increase with the round number. The i^{th} round of execution is divided into i time slots. At the beginning of each round, nodes

set the probability of transmission as $1/2$ and with each time slot in that round, the probability of transmission is reduced by half. The algorithm terminates as soon as a time slot occurs in which exactly a single node transmits.

For the networks where the number of radio nodes present is not known, Nakano et al. [11] have also proposed three different algorithms for leader election. The first algorithm proposed by them terminates with probability exceeding $1 - 1/f$ in $O((\log^2 n) + (\log^2 f))$ time slots. One of our randomized leader election algorithm is derived from this algorithm.

3 System Model

We consider leader election problem for one hop radio network with weak radio model. We term the set of all the channels as universal channel set, A_{univ} . The nodes can communicate on only a subset of the channels of the global channel set. So two nodes R_i and R_j would be termed as neighbors only if there exists some channel C_k such that both R_i and R_j can communicate on C_k . We assume that the nodes have been assigned unique labels from the set: $\{1, 2, 3, \dots, N\}$. In this paper we give a deterministic and a randomized algorithm for leader election in multi channel networks. Unless otherwise stated, we assume that all nodes are aware of the global channel set and the size of the label space. We use the following notation to describe our algorithms:

- M = Number of channels in the universal channel set
- A_{univ} = Universal channel set
- N = Size of the label space assigned to the nodes
- C_i = Channel i from the universal channel set A_{univ}
- R_i = Radio node with label i
- A_i = Set of channels available to node R_i , if R_i is present in the network
- n_{max} = Maximum number of nodes present on any channel

Observe that N acts as an upper bound on n_{max} . However, in practice, n_{max} may be much smaller than N . Also, observe that there may be one or more channels on which no node can operate. Clearly, a leader cannot be elected for such channels. Therefore we call a channel *relevant* if there are one or more nodes in the network that can operate on that channel.

4 Deterministic Algorithm for Leader Election

In this section, we propose a deterministic leader election algorithm which runs in multiple phases. With each phase, on each channel, we reduce the label space of the competing nodes by a factor of 2. Initially, the label space of the nodes competing to become leader is N for each channel. When $\log N$ phases have completed execution, the label space is reduced to a unit size for each channel. Also, we ensure that for every channel, there is exactly one node with that label in the entire network. This unique node is then elected as the leader of the corresponding channel.

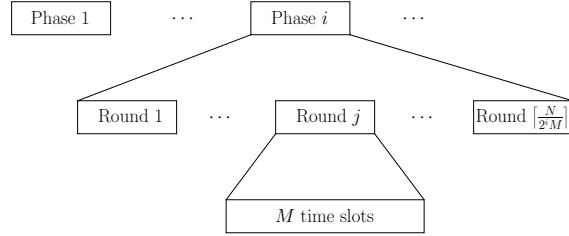


Fig. 1. Phases and rounds

Figure 1 gives the high level overview of the working of the algorithm. The execution of the algorithm is divided into $\log N$ phases and each phase is further divided into k rounds where k depends on the label space size at the beginning of the round. Further, each round takes M time slots to complete.

Working within a phase: Let us assume that at the beginning of the phase, the size of the label space of the competing nodes is X . So we divide this label space into groups of size $2M$. There are $\lceil \frac{X}{2M} \rceil$ such groups. We then further divide the execution of the phase into $\lceil \frac{X}{2M} \rceil$ rounds.

In each round, the algorithm handles one block of label space of size $2M$. In each round, we reduce the size of label space of competing nodes from $2M$ to M . So if at the start of the round, all the label space was occupied by nodes such that there were up to $2M$ nodes competing to become leader, we ensure that when the execution of the round terminates, out of the $2M$ nodes at most M nodes are still qualified for being elected as leaders. Thereafter, only these M nodes, get the chance to participate in the leader election process in the next phase. The remaining M nodes are no longer eligible for leader election on that channel.

However, if at the beginning of the phase, number of competing nodes is less than $2M$, then the phase would complete in a single round. Note that in this case as well, the algorithm would be able to reduce the label space size by a factor of 2.

Working of a round: At the beginning of each round, we group the $2M$ nodes into pairs of two and then these M node-pairs can resolve priority between themselves on all the channels in M time slots. Only the node with the higher priority, gets the chance to compete for leader election on that channel in the next round.

Table 1 shows the working of the protocol within each round. We make use of FDMA so as to reduce the number of time slots required to complete the execution of one round. This helps us reduce the time complexity of the algorithm. Also, note that the schedule is designed in such a way that all the M node-pairs get the chance to decide priority between themselves on every channel. The table is shown for the case when the range of labels of the nodes in the group is from 1

Table 1. Shows the execution of the algorithm within a round when label space of participating nodes is $\{1, 2, \dots, 2M\}$. The addition is done modulo M . In case there are some labels with no nodes, then the time slots corresponding to those labels remain blank.

Channel Time Slot	C_1	C_2	...	C_j	...	C_{M-1}	C_M
1	1, 2	3, 4		$2j - 1, 2j$		$2M - 3, 2M - 2$	$2M - 1, 2M$
2	$2M - 1, 2M$	1, 2		$2j - 3, 2j - 2$		$2M - 5, 2M - 4$	$2M - 3, 2M - 2$
i	$2M - 2i + 3, 2M - 2i + 4$	$2M - 2i + 5, 2M - 2i + 6$		$2M - 2i + 2j + 1, 2M - 2i + 2j + 2$		$2M - 2i - 1, 2M - 2i$	$2M - 2i + 1, 2M - 2i + 2$
$M - 1$	5, 6	7, 8		$2j + 3, 2j + 4$		1, 2	3, 4
M	3, 4	5, 6		$2j + 1, 2j + 2$		$2M - 1, 2M$	1, 2

to $2M$. However, if the participating nodes have labels in a different range (say $x + 1$ to $x + 2M$), then in order to figure out the transmission schedule for them, we first need to change their range temporarily from 1 to $2M$. This can be done trivially by subtracting a suitable number (x in this case) from all the labels.

So from the table we can see that for example in slot 1, nodes belonging to pair 1, *viz.* nodes 1 and 2, decide priority between themselves on channel C_1 . Also at the same time the nodes belonging to pair 2, *viz.* nodes 3 and 4, decide priority on channel C_2 .

However, if at the beginning of the round, the number of nodes competing to become leader on the channel is less than $2M$, the time taken to run the round is still M time slots.

To decide priority between two nodes (R_i and R_j) in a single time slot on some channel (C_k), we use the short routine given in Algorithm 1. The node which has higher priority moves to the next phase on channel C_k . Note that if there is no node with either of the labels, then the time slot goes empty and none of them is able to move to the next phase.

Also the nodes which move to next phase reduce their *label* by a factor of 2 and if it comes out to be fractional it is rounded up. This helps us to reduce the size of the label space by 2 with each phase. Note that the way node-pairs are formed, it is guaranteed that on any channel, two nodes with same label will never compete to become the leader. Since the size of the label space would be smaller in next phase, the number of rounds in the next phase would also reduce by a factor of 2. (Since number of rounds in a phase is proportional to the size of the label space at the beginning of the phase). However this trend continues only as long as the number of competing nodes is more than $2M$. After that, each phase contains exactly one round. We will now prove the correctness of the algorithm.

Algorithm 1. `decide_priority`

```

if  $R_i.exists$  and  $C_k \in A_i$  then
   $R_i$  sends beacon on  $C_k$ ,  $R_i$  knows it has higher priority between  $R_i$  and  $R_j$ .
  if  $R_j.exists$  and  $C_k \in A_j$  then
     $R_j$  receives beacon from  $R_i$ .  $R_j$  knows  $R_i$  has higher priority.
  end if
else if  $(\neg R_i.exists$  or  $C_k \notin A_i)$  and  $(R_j.exists$  and  $C_k \in A_j)$  then
   $R_j$  does not receive any beacon, then  $R_j$  has higher priority between  $R_i$  and  $R_j$ 
else if  $(\neg R_i.exists$  or  $C_k \notin A_i)$  and  $(\neg R_j.exists$  or  $C_k \notin A_j)$  then
  None of the nodes can communicate on  $C_k$ . So none of them move to the next
  phase.
end if

```

Lemma 1. *On any channel, no two nodes which are competing to become leader on that channel can have the same label.*

Lemma 2. *Leader is elected for all the channels wherever possible.*

Lemma 3. *For each channel, at most one leader is elected.*

Lemma 4. *The number of rounds in phase i is $\lceil \frac{N}{2^i \times M} \rceil$.*

Theorem 1. *The deterministic algorithm takes at most $2N + M \lceil \log M \rceil$ time slots for completion.*

Proof. Phase i has $\lceil \frac{N}{2^i \times M} \rceil$ rounds and each round takes M time slots for execution. Therefore the time required for completing Phase i would be $\lceil \frac{N}{2^i \times M} \rceil \times M$. Hence, total time required would be:

$$\sum_{i=1}^X \left\lceil \frac{N}{2^i \times M} \right\rceil \times M \quad (1)$$

where X is the number of phases in the execution. For values of N of the form $M \times 2^t$ (where $t \in \mathbb{N}$), $X = t + \lceil \log M \rceil$. The first t phases reduce the label space size from N to M and the remaining $\lceil \log M \rceil$ phases reduce the label space size from M to 1. It can be verified that this summation evaluates to $N - M + M \lceil \log M \rceil$. For general values of N with $N > M$ which are not of this form, we pick the smallest integer N' of the form $M \times 2^t$ (where $t \in \mathbb{N}$) that is at least N . This new integer N' is then considered as the new N . The time required for completion would be $N' - M + M \lceil \log M \rceil$ time slots. Since $N' \leq 2N$, the execution of the deterministic protocol takes at most $2N - M + M \lceil \log M \rceil$ time slots.

Finally, for values of N and M such that $N \leq M$, it can be shown that summation simplifies to $M \lceil \log N \rceil$, which is at most $M \lceil \log M \rceil$. \square

Table 2. Description of the execution of the protocol over the multiple phases. Column 2 shows the range of original labels possible for the nodes having the corresponding labels in Column 1 during the particular phase. (a) Phase 1 execution: Six labels (or 3 rows of the table) are processed in 1st round of execution. Only 4 labels are processed in the 2nd round. (b) Phase 2 execution, (c) Phase 3 execution, and (d) Phase 4 execution.

Label	Original Label	C_1	C_2	C_3
1 & 2	1 - 2	1		
3 & 4	3 - 4	3	3	
5 & 6	5 - 6		6	
7 & 8	7 - 8			7
9 & 10	9 - 10			9

(a)

Label	Original Label	C_1	C_2	C_3
1 & 2	1 - 4	1	3	
3 & 4	5 - 8		6	7
5	9 - 10			9

(b)

Label	Original Label	C_1	C_2	C_3
1 & 2	1 - 8	1	3	7
3	9 - 10			9

(c)

Label	Original Label	C_1	C_2	C_3
1 & 2	1 - 10	1	3	7

(d)

Example: We now show the working of the deterministic algorithm through an example. Let us assume that there are 3 channels and 5 nodes in the network. The initial size of label space is 10. The channel availability set of the nodes is: $A_1 = \{C_1\}$, $A_3 = \{C_1, C_2\}$, $A_6 = \{C_2\}$, $A_7 = \{C_3\}$, $A_9 = \{C_3\}$.

During the first phase, we group the label space into two groups of size 6 and 4 respectively. The transmissions during phase 1 have been given in Table 2(a). This phase takes 2 rounds to complete. We have shown the execution during both the rounds of Phase 1. Since during the phase 1, labels of the nodes are same as their original labels, therefore the values in column 1 and 2 are same. The values in the cells are the original labels of the node which had higher priority during the execution of *decide_priority()* on that channel. For example, R_1 can communicate on channel 1, therefore it has higher priority over R_2 (which does not exist in this case) on this channel.

Similarly, when the second phase terminates, the label space size would reduce to 3 (see Table 2(b)). This time the phase execution would complete in a single round. Since the node labels have now reduced by a factor of 2, therefore a node which has label 2 would actually have its original label as 3 or 4. Also note that even though both the nodes R_1 and R_3 were able to move to Phase 2 on channel 1, still only R_1 would move to Phase 3 as given by *decide_priority()*.

In the 3rd phase, the label space size would reduce to 2. This time also the phase execution would complete in a single round. (see Table 2(c)). On termination of the 4th phase, the label space size would reduce to 1. The execution proceeds in a single round as shown in Table 2(d).

Total rounds required for the execution (over all the phases) = 2 + 1 + 1 + 1 = 5. Each round takes 3 time slots to complete. Hence total execution time is

15 time slots. Note that if we had elected leaders the trivial way, the total time slots required would have been $NM = 10 * 3 = 30$ time slots.

5 Randomized Algorithms for Leader Election

In this section, we propose two different randomized algorithms for leader election. Both the algorithms work even when an upper bound on the number of nodes in the network is *not known*. We refer to the first algorithm as *Rand-Elect* and to the second as *Fast-Rand-Elect*.

5.1 Algorithm *Rand-Elect*

This algorithm for multi-channel leader election uses the algorithm proposed by Nakano and Olariu in [11] as a subroutine. Readers are referred to [11] for a detailed description. Here, we present a short overview of their algorithm. Their algorithm proceeds in multiple phases. At the beginning of each phase, nodes start with probability of transmission as $1/2$, and with each slot within the phase, the probability of transmission is reduced by a factor of 2. The number of time slots in a phase are gradually incremented by one. Therefore, phase i consists of i time slots. And, in time slot j of phase i , nodes transmit with probability $1/2^j$ (and listen with probability $1 - 1/2^j$). At the beginning of next phase, the probability of transmission is again reset to $1/2$. This pattern is continued until some time slot occurs in which *exactly one* node transmits and all other nodes listen. As soon as that happens, the unique node that transmitted in this time slot is elected as the leader of the network. Nakano and Olariu's algorithm guarantees that a leader is elected with probability at least $1 - \frac{1}{f}$ within $O(\log^2 n + \log^2 f)$ time slots, where n is the actual number of nodes present in the network [11].

To solve the leader election problem for multiple channels, we propose to run M instances of their algorithm on each of the M channels *concurrently*. The multi-channel leader election algorithm executes in rounds. Each round consists of M time slots. In the i^{th} time slot of every round, we simulate a single step of the i^{th} instance of the algorithm, which is running on channel C_i . Observe that only nodes with channel C_i in their channel availability set participate in the i^{th} instance of the algorithm.

Theorem 2. *Let n_{\max} denote the maximum number of nodes that can be present on any channel in the network. Then *Rand-Elect* elects leader for all relevant channels within $O(M \log^2(n_{\max}) + M \log^2(Mf))$ time slots with probability at least $1 - \frac{1}{f}$.*

Proof. Since $(1 - \frac{1}{Mf})^M \geq 1 - \frac{1}{f}$, therefore in order to elect leaders for all relevant channels with probability exceeding $1 - \frac{1}{f}$, it is sufficient to ensure that the probability of leader election on individual channels exceeds $1 - \frac{1}{Mf}$. Now, the i^{th} instance of Nakano and Olariu's algorithm guarantees leader election on

channel C_i with probability at least $1 - \frac{1}{Mf}$ within $O(\log^2(n_i) + \log^2(Mf))$ steps, where n_i is the number of nodes that can operate on channel C_i . Since one round simulates one step of each instance of Nakano and Olariu’s algorithm, *Rand-Elect* guarantees leader election for all relevant channels with probability at least $1 - \frac{1}{f}$ within $O(M \log^2(n_{\max}) + M \log^2(Mf))$ time slots. Hence the result. \square

5.2 Algorithm *Fast-Rand-Elect*

We now present another randomized leader election algorithm, which we call as *Fast-Rand-Elect*, that is faster than *Rand-Elect* in many cases. Before we describe the working of the new algorithm, we first describe a leader election algorithm (*Alg-Known-Size*) for single channel networks when the number of nodes present is known beforehand. The pseudo code for *Alg-Known-Size* is given in Algorithm 2 where S is the number of nodes present in the network and T is the maximum number of time slots for which the algorithm can run. In the pseudo-code, t denotes the sequence number of the current time slot.

Algorithm 2. *Alg-Known-Size*(S, T)

```

t ← 1
repeat
    transmit with probability  $\frac{1}{S}$ 
    t ← t + 1
until exactly one node transmits or t > T
    
```

It can be shown that *Alg-Known-Size* guarantees leader election with probability at least $1 - \frac{1}{e^{T/4}}$ [15], [16], [11]. Now, suppose we wish to elect a leader for all relevant channels in a multi-channel network with probability at least $1 - \frac{1}{f}$. Clearly, it is sufficient to ensure that a leader is elected for each relevant channel with probability at least $1 - \frac{1}{Mf}$ because $(1 - \frac{1}{Mf})^M \geq 1 - \frac{1}{f}$.

To elect leaders for all relevant channels in a multi-channel network when the network size (or even an upper bound on network size) is *not known*, we run multiple instances of *Alg-Known-Size* on each of the channels one-by-one with geometrically increasing values of network size. The pseudo-code of the algorithm *Fast-Rand-Elect* is given in Algorithm 3. Specifically, the execution of *Fast-Rand-Elect* is divided into multiple rounds. In round x , we run an instance of *Alg-Known-Size* on each of the channels one-by-one using an estimate of 2^x for network size. Again, as before, only those nodes in the network that can operate on channel C_i participate in the instance of *Alg-Known-Size* running on channel C_i . In the pseudo-code, i denotes the channel number and x denotes the current round number.

We now prove the correctness of the algorithm.

Lemma 5. *Let n_i denote the number of nodes in the network that can operate on channel C_i with $n_i \geq 1$. Then, by the end of $\lceil \log(n_i) \rceil$ rounds, *Fast-Rand-Elect* elects a leader on channel C_i with probability exceeding $1 - \frac{1}{Mf}$.*

Algorithm 3. *Fast-Rand-Elect* for multi-channel networks

```

 $x \leftarrow 1$ 
loop
  for  $i = 1$  to  $M$  do
    run an instance of Alg-Known-Size( $2^x, 8 \ln(Mf)$ ) on channel  $C_i$ 
  end for
   $x \leftarrow x + 1$ 
end loop

```

Theorem 3. Let n_{\max} denote the maximum number of nodes present on any channel in the network. Then *Fast-Rand-Elect* elects a leader on all relevant channels with probability exceeding $1 - \frac{1}{f}$ within $O(M \log(n_{\max}) \log(Mf))$ time slots.

Proof. From Lemma 5, *Fast-Rand-Elect* elects a leader on a relevant channel with probability at least $1 - \frac{1}{Mf}$ within $O(\log(n_{\max}))$ rounds. Each round consists of $O(M \log(Mf))$ time slots. Therefore *Fast-Rand-Elect* elects a leader on a relevant channel with probability at least $1 - \frac{1}{Mf}$ within $O(M \log(n_{\max}) \log(Mf))$ time slots. This, in turn, implies that *Fast-Rand-Elect* elects a leader on all relevant channels with probability at least $1 - \frac{1}{f}$ within $O(M \log(n_{\max}) \log(Mf))$ time slots. \square

Observe that *Fast-Rand-Elect* has better asymptotic time complexity than *Rand-Elect* if $\log(n_{\max})$ is either $o(\log(Mf))$ or $\omega(\log(Mf))$. In other words, $\log(n_{\max}) \notin \Theta(\log(Mf))$. In case $\log(n_{\max}) = \Theta(\log(Mf))$, the two algorithms have the same asymptotic time complexity. On the other hand, *Fast-Rand-Elect* requires a user to fix the success probability (given by $1 - 1/f$) a priori.

6 Conclusion

In this paper, we have presented three leader election algorithms for multi channel radio networks with no collision detection. The deterministic algorithm guarantees leader election for all relevant channels (available to one or more nodes in the network) and takes at most $2N + M \lceil \log M \rceil$ time slots, which is a considerable improvement over the trivial algorithm which takes NM time slots for completion. The two randomized algorithms guarantee leader election for all relevant channels with probability at least $1 - \frac{1}{f}$ within $O(M \log^2(n_{\max}) + M \log^2(Mf))$ and $O(M \log(n_{\max}) \log(Mf))$ time slots, respectively.

Note that, when $n_{\max} = \Theta(M)$, the deterministic algorithm has time-complexity of $O(M \log M)$, which is asymptotically better than that of the randomized algorithms. Moreover, with a randomized algorithm, there is always a non-zero probability that a leader may not be elected for one or more of the relevant channels. However, the deterministic algorithm guarantees that a leader is elected for all the relevant channels with 100% confidence.

References

1. Akyildiz, I.F., Lee, W.Y., Vuran, M.C., Mohanty, S.: Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey. *Computer Networks* 50(13), 2127–2159 (2006)
2. Mitola, J.: *Cognitive radio: An Integrated Agent Architecture for Software Defined Radio*. PhD thesis, Royal Institute of Technology (2000)
3. Perkins, C.E., Belding-Royer, E.M.: Ad-hoc on-demand distance vector routing. In: *Proceedings of the 2nd Workshop on Mobile Computing Systems and Applications (WMCSA)*, pp. 90–100. IEEE Computer Society, Los Alamitos (1999)
4. DeCleene, B., Dondeti, L., Griffin, S., Hardjono, T., Kiwior, D., Kurose, J., Towsley, D., Vasudevan, S., Zhang, C.: Secure group communication for wireless networks. In: *Proceedings of the IEEE Military Communications Conference (MILCOM)*, pp. 113–117 (2001)
5. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: *Proceedings of the Annual Hawaii International Conference on System Sciences (HICSS)* (2000)
6. Mittal, N., Krishnamurthy, S., Chandrasekaran, R., Venkatesan, S.: A Fast Deterministic Algorithm for Neighbor Discovery in Multi-Channel Cognitive Radio Networks. Technical Report UTDCS-14-07, The University of Texas at Dallas (2007)
7. Krishnamurthy, S., Thoppian, M.R., Kuppa, S., Chandrasekaran, R., Mittal, N., Venkatesan, S., Prakash, R.: Time-efficient distributed layer-2 auto-configuration for cognitive radio networks. *Computer Networks* 52(4), 831–849 (2008)
8. Bansal, T., Mittal, N., Venkatesan, S.: Leader Election Algorithms for Multi-Channel Wireless Networks. Technical Report UTDCS-19-08, The University of Texas at Dallas (2008)
9. Jurdzinski, T., Kutylowski, M., Zatoptionski, J.: Weak communication in single-hop radio networks: Adjusting algorithms to industrial standards. *Concurrency and Computation: Practice and Experience* 15(11-12), 1117–1131 (2003)
10. Jurdzinski, T., Kutylowski, M., Zatoptionski, J.: Efficient algorithms for leader election in radio networks. In: *Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 51–57 (2002)
11. Nakano, K., Olariu, S.: Randomized leader election protocols in radio networks with no collision detection. In: Lee, D.T., Teng, S.-H. (eds.) *ISAAC 2000*. LNCS, vol. 1969, pp. 362–373. Springer, Heidelberg (2000)
12. Nakano, K., Olariu, S.: Uniform leader election protocols for radio networks. *IEEE Trans. Parallel Distrib. Syst.* 13(5), 516–526 (2002)
13. Bordim, J.L., Ito, Y., Nakano, K.: An energy efficient leader election protocol for radio network with a single transceiver. *IEICE Transactions* 89-A(5), 1355–1361 (2006)
14. Nakano, K., Olariu, S.: A survey on leader election protocols for radio networks. In: *Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN)*, pp. 63–68. IEEE Computer Society, Los Alamitos (2002)
15. Metcalfe, R., Boggs, D.: Ethernet: Distributed packet switching for local computer networks. *Commun. ACM* 19(7), 395–404 (1976)
16. Hayashi, T., Nakano, K., Olariu, S.: Randomized initialization protocols for packet radio networks. In: *Proceedings of the 13th International Parallel and Processing Symposium (IPPS)*, pp. 544–548. IEEE Computer Society, Los Alamitos (1999)