Energy Efficient Naming in Beeping Networks

Ny Aina Andriambolamalala[1] and Vlady Ravelomanana[2]

[1] IRIF — University Paris Diderot — France. Ny-Aina.Andriambolamalala@irif.fr
[2] IRIF UMR CNRS 8243 — University Paris Diderot — France vlad@irif.fr

Abstract. The Beeping network is a distributed communication model in which all devices can communicate by sending or receiving only $1-bit$ messages (beep). In this paper, we focus on resolving two fundamental distributed computing problems, the *naming* and the *counting* problems. As most of the time, devices are battery-powered, managing energy is important in order to successfully terminate any distributed task. In a distributed beeping network with n stations, we design energy efficient randomized algorithms with an optimal running time of $O(n \log n)$ and optimal $O(\log n)$ energy complexity.

Keywords: Distributed · initialization· naming · energy· optimal· beep

1 Introduction

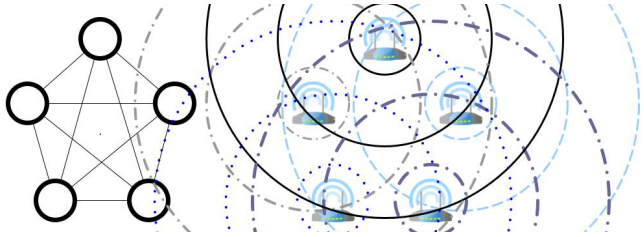
The beeping model, introduced by Cornejo and Kuhn in 2010 [8], makes little demands on the devices which need only be able to do carrier-sensing, differentiating between silence and the presence of a jamming signal on the network (considered as $1-bit$ message or one beep). It is assumed that the devices have unbounded local power computation. They note that carrier-sensing can typically be done much more reliably and requires significantly less energy and other resources than message-sending models. Minimizing such energy consumption per node arises as nodes are battery-powered. Since sending or receiving messages costs more energy than internal computations, energy consumption is measured as the maximal number of time slots during which a node is awake (beeps or listens to the network) [6, 20, 17, 14, 15, 14, 26]. In distributed computing, in order to have a more realistic model, it is usually assumed that the nodes have no prior information about the topology of the network and that the nodes are initially indistinguishable (have no identifier denoted ID). With such an assumptions, it is more difficult to design efficient and elegant distributed algorithms. To break such initial symmetry, researchers designed various protocols such as leader election ([6, 16, 13, 10, 11, 17, 24, 22, 15]) Maximal Independent Set ([1, 25]) and naming protocols also known as initialization algorithms ([19, 12, 20, 2, 7]). In this paper, we consider the naming problem on single-hop[1] beeping networks which consists of assigning a unique label $\ell \in \{1, 2, ...n\}$ to each of the n nodes of the network. In the general case, when the nodes are initially

[1] The underlying graph of the network is a complete graph.

indistinguishable and no node knows n, we design an energy optimal randomized naming algorithm succeeding in $O(n \log n)$ time slots with high probability [2] (*w.h.p.*), with no node being awake for more than $O(\log n)$ time slots. We start by presenting the basic brick of our algorithms, naming M nodes ($M \leq n$) in $O(M \log n)$ time slots with $O(M + \log n)$ energy (Section 2). We consider the case where all nodes are initially indistinguishable and do not know n (Section 3). This latter algorithm can be adapted to solve the counting problem, which consists of assigning to all nodes a common value representing their exact number (Section 3). Thereafter, we use derandomization techniques to adapt our naming algorithm in order to have a deterministic algorithm even if n is not known beforehand (Section 4). As customary in deterministic settings, we assume that the nodes have unique ID $\in \{1, 2, ...N\}$, where N is an upper bound of n. W.h.p., this deterministic version terminates with no node being awake for more than $O(\log n)$ time slots. Finally, we prove a lower bound of $\Omega(\log n)$ on the energy complexity for the naming problem in beeping networks (Section 5).

1.1 The models

In a single-hop beeping network, nodes communicate with each other via a shared beeping channel. As shown in the figure below, this can be used for modeling an ad hoc network where all nodes are in each other's communication range, the nodes can send $1 - bit$ messages and do a carrier sensing in order to detect any transmission.



In each synchronous discrete time slot, each node independently decides whether to transmit a beep, to listen to the network or to remain idle (asleep). Only listening nodes can receive the state of the common channel which can be, Beep if at least one node is transmitting or Null when no node transmits. This model is also called BL or Beep Listen model. There are many variants of beeping model but in this paper, we use in general the BL except for the randomized counting protocol where we will use the $B_{\text{CD}}L$ model (Beep with Collision Detection Listen) where transmitters can detect collision [1, 25].

1.2 Related works and new results

As a fundamental distributed computing problem [19], many results exist for the naming problem. In [12], Hayashi, Nakano and Olariu presented a $O(n)$ time

[2] Event ε_n occurs with high probability if $\mathbb{P}[\varepsilon_n] \geq 1 - \frac{1}{n^c}$ for any constant $c > 0$

randomized protocol for radio networks with collision detection (RNCD). Later, Bordim, Cui, Hayashi, Nakano and Olariu [2] presented an algorithm terminating *w.h.p.* in $O(n)$ time slots, and $O(\log n)$ energy. In [21], for radio network with no collision detection (RNnoCD), Nakano and Olariu designed a protocol terminating in $O(n)$ time slots *w.h.p.* with $O(\log \log n)$ energy. The results on beeping model appeared very recently when Chlebus, De Marco and Talo [7] presented their algorithm terminating in $O(n \log n)$ *w.h.p.* for the BL model and provided $\Omega(n \log n)$ lower bound on time complexity. Moreover, Casteigts, Métivier, Robson and Zemmari [4] presented a counting algorithm for the $B_{\mathrm{CD}}L$ model terminating in $O(n)$ time slots *w.h.p.* They noticed that adapting their algorithm to BL model will cost a logarithmic slowdown in time complexity. The following table compares existing results with ours.

Existing results			
Problem and Model	Time complexity	Energy complexity	Succeed with probability
Randomized naming in RN NoCD [21], n known	$O(n)$	$O(\log \log n)$	$1 - O(\frac{1}{n})$
Randomized naming in BL [7], n known	$\Theta(n \log n)$	-	$1 - O(\frac{1}{n})$
Randomized Counting in $B_{\mathrm{CD}}L$ [4]	$O(n)$	-	$1 - O(\frac{1}{n})$
Our results			
Randomized naming in BL network Indistinguishable nodes, n unknown, Theorem 2	$O(n \log n)$	$\Theta(\log n)$	$1 - O(\frac{1}{n^c})$ $c > 0$
Unconditional Deterministic naming in BL Unique ID $\in \{1, N\}$, n unknown, Theorem 1	$O(n \log n)$	$O(n)$	-
Derandomized Deterministic naming in BL Unique ID $\in \{1, N\}$, n unknown, Theorem 4	$O(n \log n)$	$O(\log n)$	$1 - O(\frac{1}{n^c})$ $c > 0$
Randomized Counting in BL Theorem 3	$O(n \log n)$	$\Theta(\log n)$	$1 - O(\frac{1}{n^c})$ $c > 0$

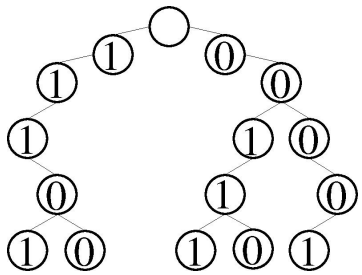
2 New approach : deterministic naming of M nodes

In this section, for the sake of clarity, let N be a polynomial upper bound of n ($N = n^c, c > 1$ and c is constant), known in advance by the nodes and each node has a unique identifier denoted ID $\in \{1, 2, ...N\}$. We will see in the following sections that N can be randomly approximated by the nodes even if no node knows n and that the nodes can randomly generate unique ID *w.h.p.* if they are initially indistinguishable.

If M nodes ($M \leq n$) hold such unique ID, they first encode their ID into binary code-word denoted CID = [CID[1] CID[2]... CID[$\lceil \log_2 N \rceil$]] such that CID[i] $\in \{0, 1\}$ (CID[1] corresponds to $2^{\lceil \log_2 N \rceil}$ and CID[$\lceil \log_2 N \rceil$] corresponds to 2^0). Each participant has to send its CID bit by bit during $\lceil \log_2 N \rceil = O(\log n)$ time slots in order to know if it has the maximal ID of all participants. During such $O(\log n)$ time slots, each node detecting that any of its neighbors has a higher

ID gets eliminated and at the end, the unique node holding the maximal ID remains active and gets the next available label.
As a consequence, such an algorithm can be subdivided into M deterministic seasons $S_1, S_2, ...S_M$ (note that nodes don't know M). In one season S_j, each node sends its CID bit by bit during $\lceil \log_2 N \rceil$ steps $t_1, t_2, ...t_{\lceil \log_2 N \rceil}$ (corresponding to CID[1], CID[2], ... CID[$\lceil \log_2 N \rceil$]). One of such steps, t_i consists of two communication time slots t_{i_0} and $t_{i_1} = t_{i_0} + 1$. If a node s has CID[i] = 0, s beeps at t_{i_0} and listens to the network at t_{i_1}. Respectively, if s has CID[i] = 1, s listens to the network at t_{i_0} and beeps at t_{i_1}. Each node knows if at least one of them has CID[i] = 1. In this case, each node s having CID[i] = 0 becomes inactive until the next season S_{j+1} (s is eliminated for the current season). At the end of season S_j, the last active node takes the label j. By looping these computations until no node remains unlabeled, this method produces a naming algorithm terminating in $O(M \log n)$ time slots.
For a better comprehension, we represent the algorithm as a binary tree as done in [9]. One node of the tree represents any bit CID[i] of any device. Its root is the first bit of CID and the leaves represent the last bit.

In this figure, we can see a simple example for 5 nodes having ID $\in \{61, 60, 7, 6, 1\}$. The rightmost node s represents the device having CID = [00001]. On the first step t_1 of the season S_1, all nodes having CID[1] = 1 go left on the tree and nodes having CID[1] = 0 go right. Then, all nodes repeat the same for CID[2] on season S_2 and so on. We can see that s wakes up 5 times on each node of the tree.



To simplify all these computations, we define Test(i) protocol as a subroutine of our algorithms. It takes step number 'i' as parameter, outputs each node with a state $\in \{$Eliminated, Active$\}$ as follows. For a node s calling Test(i) at any step t_i, if s has CID[i] = 0, s beeps at t_{i_0} and listens to the network at $t_{i_1} = t_{i_0} + 1$. If s hears beep at t_{i_1}, Test(i) returns Eliminated. If s has CID[i] = 1, s listens to the network at t_{i_0} and beeps at t_{i_1}. If s hears beep at t_{i_0}, Test(i) returns Active. So in step t_i, all nodes receiving Test(i) = Eliminated, becomes inactive until the next season. The last active node in the current season takes a label.

Energy optimization principle: The latter algorithm is not energy efficient because all nodes have to be awake during the whole $O(M \log n)$ time slots. To improve such energy consumption, we remark that each node s must be awake only during two specific sets of steps in order to know if any of its neighbors has a higher ID. Thus, we introduce these two kinds of steps as *steps to notify (*STN*)* and *steps to listen (*STL*)*. As binary code-word are unique, there is at least one bit differentiating any two code-words of any pair of nodes s_1 and s_2.

The main goal of using STN and STL is to wake up only at these important bits (steps).

Algorithm 1. TEST(i) at any node s

Input : A unique code-word CID,the current step number i
Output: ACTIVE or ELIMINATED

```
1  if CID[i] = 0 then
2  |  s beeps at t_{i_0} and listens at t_{i_1} = t_{i_0} + 1
3  |  if s hears beep at t_{i_1} then
4  |  |  return ELIMINATED
5  |  end
6  else
7  |  s listens at t_{i_0}
8  |  if s hears beep at t_{i_0} then
9  |  |  return ACTIVE
10 |  end
11 |  s beeps at t_{i_1}
12 end
```

Definition 1 (Step To Listen : STL). *A* STL *is one step t_i recorded by the node s during any season S_j, on which s has to wake up, during all the next seasons $S_{j+1}, ...S_M$ in order to listen at t_{i_0}. Nodes waking up at any step $t_i \in$ STL may not sleep after t_i. At any step t_i of season S_j, a node s_2, having* TEST*(i)* = ELIMINATED *records i into* STL *because on the next season, s_2 has to wake up at t_i in order to verify if it is still eliminated at this step.*

Definition 2 (Steps To Notify : STN). *A* STN *is a set of steps $\{t_i, t_k, ...\}$ recorded by the node s during any season S_j, on which s has to wake up, during all the next seasons $S_{j+1}, ...S_M$, beeping at t_{i_1}. s sleeps after any step $t_i \in$ STN. During any step t_i of season S_j, a node s_1 receiving* TEST*(i)* = ACTIVE *(has bit* CID$[i] = 1$*) knows there is at least one node s_2 having* CID$[i] = 0$*. s_1 saves i into* STN *because on the next season, if s_1 is active, it has to wake up on t_i in order to notify s_2 that s_2 is still eliminated at this step. When a node s_1 adds any step t_i to* STN*, it means that s_1 has no more active neighbor holding* CID$[k] = 1, k > i$*. Thus, s_1 has to empty* STL*.*

Description of Algorithm 2, the energy efficient algorithm: During the first season S_1, all nodes are initially awake and start to send their code-word CID bit by bit. In any step t_i, if a node s_1 have TEST(i) = ELIMINATED, s_1 adds i into STL and sleeps until the next season. A node s_2 with TEST(i) = ACTIVE adds i into STN, empties STL and moves to t_{i+1}. A node s remaining awake at the end of S_1 sets $\ell = 1$, empties STN and STL and sleeps. We can generalize the algorithm for any season S_j. As at the end of the previous season S_{j-1}, all nodes are sleeping, a node s_3 wakes up only at the first step t_i found in its STL or STN. If such $i \in$ STN, then s_3 sleeps before moving on t_{i+1}. But if $i \in$ STL, s_3 acts as in season S_1: if s_3 has TEST(i) = ELIMINATED, then s_3 adds

i into STL and sleeps until the next season. s_3 stays awake and moves on t_{i+1} otherwise. At the end of season S_j, the last remaining awake node sets $\ell = j$, empties STN and STL and sleeps.

Algorithm 2. DeterministicNaming(N) on any node s

Input : Upper bound N of n, unique ID $\in \{1, 2, ...N\}$
Output: Node s has unique label $\ell \in \{1, 2, ...M\}$

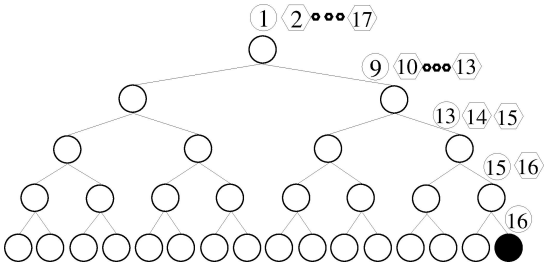
s encodes ID into binary code-word CID $= \{0,1\}^{\lceil \log_2 N \rceil}$,
s sets $\ell \leftarrow 0$, STL $\leftarrow$ Null, STN $\leftarrow$ Null, $S \leftarrow 1$
while $\ell = 0$ **do**
  for i *from* $0 \rightarrow \lceil \log_2 N \rceil$ **do**
    if $i \in$ STL **then**
      s wakes up at t_i and **if** Test(i) = Eliminated **then**
        s sleeps
      end
    end
    if $i \in$ STN **then**
      s wakes up at t_i ,does Test(i) and sleeps
    end
    if s *is awake* **then**
      if Test(i) = Active **then**
        s adds i to STN and empties STL
      end
      if Test(i) = Eliminated **then**
        s adds i to STL and sleeps
      end
    end
  end
  if s *is awake* **then**
    s stes $\ell \leftarrow S$, empties STL, empties STN and sleeps
  end
  $S \leftarrow S + 1$
end

Lemma 1. *In single hop beeping networks of size n, there is a deterministic algorithm naming some M participating nodes in $O(M \log n)$ time slots with no node being awake for more than $O(M + \log n)$ steps.*

Proof. Algorithm 2 terminates deterministically in $M \times \lceil \log N \rceil = O(M \log n)$ time slots. In the following, let W_s be the total waking times of any node s in Algorithm 2, W_{STN} , W_{STL} and W_{other} correspond to STN total waking time, STL and other total waking times. Similarly, $(W_{STN})_{worst}$, $(W_{STL})_{worst}$ and $(W_{other})_{worst}$ are the worst waking times of all nodes. We have

$$W_s = W_{\text{STN}} + W_{\text{STL}} + W_{other} \leq (W_{\text{STN}})_{worst} + (W_{\text{STL}})_{worst} + (W_{other})_{worst}\,. \quad (1)$$

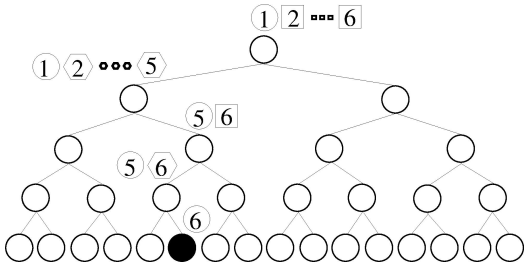
In order to find $(W_{STN})_{worst}$ and $(W_{STL})_{worst}$, we can simulate a complete binary tree to be the Tree representation of the networks devices as done in [9]. In the figures below, the hexagons represent the STL waking steps of the black node, the squares represent the STN waking steps and the circles represent the other waking steps. The number inside these shapes represents the season during which the node wakes up. For a better comprehension, we illustrate how we obtained the two following figures in Appendix 2 and Appendix 3.



a) Worst case for STL.

The node s having $(W_{STL})_{worst}$ (the black node in Figure **a)**) wakes up T times in any step t_i of STL until no other node has a higher ID. This value T is at most half of participants on t_1 and gets halved every i. We can see (by the diamonds shapes in Figure **a)**), that s wakes up $\frac{M}{2}+1$ times in season S_1.

Furthermore, as for STL, a node s wakes up at $t_i \in$ STN during as many times as the number of nodes with higher ID. I.e., half the number of participants at step t_1. This value gets halved every i and we have



b) Worst case for STN.

$$(W_{STL})_{worst} \leq \sum_{i=1}^{M} \left(\frac{M}{2^i} + 1 \right) \leq O(M) \text{ and } (W_{STN})_{worst} \leq \sum_{i=1}^{M} \left(\frac{M}{2^i} + 1 \right) \leq O(M) \tag{2}$$

A node s wakes up just once at any step t_i not in STN and STL (we can see that by the round shapes in Figure **a)** and **b)**). Hence, we have

$$(W_{other})_{worst} \leq \sum_{i=1}^{\log N} O(1) \leq O(\log N) \leq O(\log n)\,. \tag{3}$$

□

A Maple simulation is available in Appendix 1.

Theorem 1. *In single-hop beeping networks of size n, if n is known in advance by all nodes and nodes have unique* ID $\in \{1, 2, ...N\}$ *(N is an upper bound of n), there is an energy efficient deterministic naming algorithm, assigning unique label to all nodes in* $O(n \log n)$ *rounds, with no node being awake for more than* $O(n)$ *time slots.*

Proof. Applying Lemma 1 to $M = n$, we reach the desired result. □

3 Energy Efficient Randomized algorithms

We assume that the total number of nodes is unknown and that nodes are initially indistinguishable. All the nodes have to know a linear approximation u of n. This approximation problem was well studied in the distributed computing area. Brandes, Kardas, Klonowski, Pająk and Wattenhofer [3] designed a randomized linear approximation algorithm, terminating *w.h.p.* in $O(\log n)$ rounds. Our main idea is to assign a unique ID uniformly from $\{1, 2, ...u^3\}$ to all nodes and use the well known balls and bins problem for putting our n nodes into $\lceil \frac{u}{\log u} \rceil = O(\frac{n}{\log n})$ groups.

Lemma 2. *As a classical result (see for instance [23]), if n nodes randomly and uniformly choose to enter into $\lceil \frac{n}{\log n} \rceil$ groups, there is at most $4 \log n$ nodes in each group with high probability.*

Proof. The probability to enter any group G_i is $\frac{\log n}{n}$. As a consequence, if $|G_i|$ denotes the number of nodes in group G_i, then $\mathbb{E}[|G_i|] = \log n$. Hence, by mean of Chernoff bound, $|G_i| \leq 4 \log n$ with probability at least $1 - \frac{1}{n^2}$. □

We sequentially run DeterministicNaming(u^3) on each group one by one such that each group works during at most $\lceil \log u^3 \rceil = O(\log n)$ time slots to name themselves and during an extra $O(\log n)$ rounds where the last labeled node sends its label bit by bit to the next group. As DeterministicNaming(N) outputs a labeling of the M participants such that $\ell \in \{1, 2, ..M\}$, after the call of this algorithm, all nodes have to update their label such that the first label outputted by DeterministicNaming(N) in season S_1 of a group G_k must be the next available label after the last label outputted in group G_{k-1}.

In order to know if any node s has the last label of its group, we modify DeterministicNaming(N) algorithm such that a node labeled at season S_j wakes up during the entire season S_{j+1} and listens to the network to find remaining unlabeled nodes. This extra $O(\log n)$ waking time doesn't affect our $O(\log n)$ energy complexity.

Theorem 2. *In single-hop beeping networks of size n, if n is unknown by all nodes and nodes are initially indistinguishable, there is an energy efficient randomized naming algorithm, assigning a unique label to all nodes in $O(n \log n)$ w.h.p, with no node being awake for more than $O(\log n)$ time slots.*

Proof. Using DeterministicNaming(N), the latter described algorithm is quasi deterministic. As by [3], $u = \Theta(n)$, if we note T_D, the time complexity of DeterministicNaming($N = u^3$) algorithm, our naming algorithm terminates in $\lceil \frac{u}{\log u} \rceil \times T_D$ time. By Lemma 2, the number of participants is $O(\log n)$, and using Lemma 1 we get $T_D = O(\log^2 n)$, implying the $O(n \log n)$ time complexity of Algorithm 3.

Therefore, as each node s is awake only during the execution of DeterministicNaming(u^3) and $O(\log n)$ extra times for notifying the next group and checking

if s has the last label, the energy complexity is $O(\log n)$. □
By doing some adaptation, we can design an algorithm with $O(n \log n)$ time complexity and $O(\log n)$ energy complexity on this problem on single-hop BL network. □

Theorem 3. *In single-hop beeping networks of size n, if n is unknown by all nodes and nodes are initially indistinguishable, there is an energy efficient randomized counting algorithm allowing all the nodes to know the exact number of the participating nodes, terminating in $O(n \log n)$ w.h.p, with no node being awake for more than $O(\log n)$ time slots.*

Proof. If at the end of the last group $G_{\lceil \frac{u}{\log u} \rceil}$, all nodes wake up and the last labeled node sends its label bit by bit, this corresponds *w.h.p.* to the exact number of nodes on the network. □

4 Deterministic energy efficient naming algorithm

As the randomized part of our algorithm is the assignment of all nodes to $\lceil \frac{n}{\log n} \rceil$ groups of size $O(\log n)$, our goal is to do a deterministic assignment with a very small error rate. In order to do so, we use a hash function in order to map each node's ID to $\lceil \frac{n}{\log n} \rceil$ values, such that the nodes holding the same value belong to the same group.

Theorem 4. *In single-hop beeping networks of size n, if n is known in advance by all nodes and nodes have a unique* ID $\in \{1, 2, ...N\}$ *(N is an upper bound of n), there is an energy efficient deterministic naming algorithm, assigning unique label $\ell \in \{1, 2, ...n\}$ to all nodes in $O(n \log n)$, having no node being awake for more than $O(\log n)$ time slots, with probability of error less than $O\left(\frac{1}{n^c}\right)$, $c > 0$.*

Proof. Celis, Reingold, Segev and Wieder [5] construct such hashing function, by encoding integer values into binary code-word of length $O(\log n \log \log n)$, such that there is at most $O(\frac{\log n}{\log \log n})$ integers mapped to the same code-word *w.h.p.* By using such a hash function to replace randomness on our algorithm (as local computations, no message has to be sent), we prove the latter Theorem. □

5 Lower bound on energy complexity

In [7], the authors presented an $\Omega(n \log n)$ lower bound for the running time of any randomized naming algorithm. Note that in [6], Chang *et al* studied the energy complexity of leader election, approximate counting and census in several models of wireless radio networks with messages of unbounded size. In this paragraph, we prove that $O(\log n)$ time is necessary and sufficient for one node to be the unique transmitter of 1 single bit in the beep model with constant probability. Our proof uses Yao's minimax principle [27]. The idea is to adapt Liu and Prabhakaran [18] proof for the lower bound on broadcasting algorithm in radio networks. Let's recall Yao's minimax principle.

Theorem 5 (Yao's minimax principle [27] – Theorem 3)**.** *Let $0 < \lambda < 1/2$. Let $\mathcal{P}$ be a probability distribution over the set of inputs. Let $\mathcal{A}$ denote the set of all deterministic algorithms that err with probability at most 2λ over $\mathcal{P}$. For $A \in \mathcal{A}$ let $C(A, \mathcal{P})$ denote the expected running time of A over $\mathcal{P}$. Let $\mathcal{R}$ be the set of randomized algorithms that err with probability at most λ for any input, and let $\mathbb{E}(R, I)$ denote the expected running time of R on input I. Then, for all $\mathcal{P}$ and all $R \in \mathcal{R}$,*

$$\min_{A \in \mathcal{A}} C(A, P) \leq 2 \max_{I} \mathbb{E}(R, I)\,.$$

This principle offers a generic tool that permits to prove randomized lower bound by reducing this task to that of proving deterministic lower bound.

Lemma 3. *Let s be any given node of the beeping network. For any randomized algorithm, the expected running time for s to be the unique sender of exactly 1 bit is $\Omega(\log n)$.*

Proof. We prove lemma 3 on the $B_{\mathrm{CD}}L_{\mathrm{CD}}$ beeping network which is a stronger model than the harsh BL model. That is, in the following proof, all non-idle nodes (senders and listeners) will be (temporarily) able to distinguish between exactly 1 node or at least two nodes have sent the same bit. W.l.o.g., fix a given time slot t and suppose that the node s has to recognize that it is the unique sender of a bit equals to 0. Define a configuration of a network at any given time slot as the union of 4 sets of nodes: a set of idle nodes, a set of listeners, a set of nodes transmitting the bit 1 (resp. 0). Adapting the methods in [18, §3] for our purpose, we define a family $\mathcal{F}_k$ as the set of all configurations of the n nodes such that k nodes hold a 0 and $n-k$ nodes hold 1. Observe that a node holding a bit can be idle. We assign to this family the weight $\frac{c}{k \log n}$ where c is a normalization factor ($1/c = \sum_{k=1}^{n} \frac{1}{k \log n}$ and $c \sim 1$ as n is large). During the deterministic time slot t, we can suppose that j stations are awake, k stations hold 0. We observe that any station s succeeds to send its bit (equals to 0) without any collision if
- it is one of the non-idle stations and it sends a 0 and
- the $k-1$ other holders of 0 at this time are idle.
Thus the probability that a unique station is sending 0 over the union of the $\mathcal{F}_k$ (i.e. $\mathcal{F} = \cup_{k=1}^{n} \mathcal{F}_k$) is

$$c \sum_{j=1}^{n} \frac{\binom{j}{1}\binom{n-j}{k-1}}{k\binom{n}{k} \log n} = \frac{c}{\log n} \frac{j}{n} \sum_{k=1}^{n-j+1} \frac{\binom{n-j}{k-1}}{\binom{n-1}{k-1}} = O\left(\frac{1}{\log n}\right).$$

We have just shown that there is a probability distribution such that the probability that there is a unique sender of 1 bit in any one deterministic step is $O(1/\log n)$. Thus, over the probability distribution we pick, the expected running time of any deterministic algorithm that errs with probability $1 - \Omega(1)$ allowing a node to be the unique sender of 1 bit is $\Omega(\log n)$. □

Consequently we have the following lower bound.

Theorem 6. *The energy complexity of any randomized algorithm solving the naming problem with constant probability is $\Omega(\log n)$.*

Proof. To prove this, fix any given free label $\ell \in \{1, \cdots, n\}$. A node that picked that number has to signal to the others that it picks the label ℓ. Such a signal consists in $\Omega(1)$ bit(s) and at least 1 of these bits has to be the only bit present on the shared channel at some given time t. By Lemma 3, this requires $\Omega(\log n)$ transmissions to succeed with constant probability. □

Conclusion

In this paper, we focus on the naming problem in single-hop beeping networks. We start by a randomized version, when the nodes do not have any information about the size n of the network and are initially indistinguishable, terminating in optimal $O(n \log n)$ time slots *w.h.p.*, and optimal $O(\log n)$ energy complexity. We have also established that for the same task, $\Omega(\log n)$ awake time slots is necessary for any randomized algorithm with constant probability of success. Our algorithm can be used for the counting problem, returning the exact number of nodes in $O(n \log n)$ time slots, with $O(\log n)$ energy complexity. We also design an energy-efficient unconditional deterministic naming algorithm terminating in $O(n \log n)$ time slots with $O(n)$ energy. By means of derandomization, we devise an energy-efficient deterministic naming algorithm that errs with probability less than $O\left(\frac{1}{n^c}\right)$ terminating in $O(n \log n)$ time slots and $O(\log n)$ energy complexity.

References

1. Afek, Y., Alon, N., Bar-Joseph, Z., Cornejo, A., Haeupler, B., Kuhn, F.: Beeping a maximal independent set. Distributed computing **26**(4), 195–208 (2013)
2. Bordim, J.L., Cui, J., Hayashi, T., Nakano, K., Olariu, S.: Energy-efficient initialization protocols for ad-hoc radio networks. In: International Symposium on Algorithms and Computation. pp. 215–224. Springer (1999)
3. Brandes, P., Kardas, M., Klonowski, M., Pająk, D., Wattenhofer, R.: Approximating the size of a radio network in beeping model. In: International Colloquium on Structural Information and Communication Complexity. pp. 358–373. Springer (2016)
4. Casteigts, A., Métivier, Y., Robson, J.M., Zemmari, A.: Counting in one-hop beeping networks. to appear Theoretical Computer Science (2016)
5. Celis, L.E., Reingold, O., Segev, G., Wieder, U.: Balls and bins: Smaller hash families and faster evaluation. SIAM Journal on Computing **42**(3), 1030–1050 (2013)
6. Chang, Y.J., Kopelowitz, T., Pettie, S., Wang, R., Zhan, W.: Exponential separations in the energy complexity of leader election. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing. pp. 771–783. ACM (2017)
7. Chlebus, B.S., De Marco, G., Talo, M.: Naming a channel with beeps. Fundamenta Informaticae **153**(3), 199–219 (2017)
8. Cornejo, A., Kuhn, F.: Deploying wireless networks with beeps. In: International Symposium on Distributed Computing. pp. 148–162 (2010)
9. Fuchs, M., Hwang, H.K.: Dependence between external path-length and size in random tries. arXiv preprint arXiv:1604.08658 (2016)

10. Ghaffari, M., Haeupler, B.: Near optimal leader election in multi-hop radio networks. In: Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms. pp. 748–766 (2013)
11. Ghaffari, M., Lynch, N., Sastry, S.: Leader election using loneliness detection. Distributed Computing **25**(6), 427–450 (2012)
12. Hayashi, T., Nakano, K., Olariu, S.: Randomized initialization protocols for packet radio networks. In: ipps. p. 544. IEEE (1999)
13. Jurdziński, T., Kutyłowski, M., Zatopiański, J.: Efficient algorithms for leader election in radio networks. In: Proceedings of the twenty-first annual symposium on Principles of distributed computing. pp. 51–57. ACM (2002)
14. Jurdziński, T., Kutyłowski, M., Zatopiański, J.: Energy-efficient size approximation of radio networks with no collision detection. In: International Computing and Combinatorics Conference. pp. 279–289 (2002)
15. Kardas, M., Klonowski, M., Pajak, D.: Energy-efficient leader election protocols for single-hop radio networks. In: Parallel Processing (ICPP), 2013 42nd International Conference on. pp. 399–408. IEEE (2013)
16. Kutten, S., Pandurangan, G., Peleg, D., Robinson, P., Trehan, A.: Sublinear bounds for randomized leader election. In: International Conference on Distributed Computing and Networking. pp. 348–362. Springer (2013)
17. Lavault, C., Marckert, J.F., Ravelomanana, V.: Quasi-optimal energy-efficient leader election algorithms in radio networks. Journal of Information and Computation **205**(5), pages–679 (2007)
18. Liu, D., Prabhakaran, M.: On randomized broadcasting and gossiping in radio networks. In: Computing and Combinatorics, 8th Annual International Conference, COCOON 2002, Singapore, August 15-17, 2002, Proceedings. pp. 340 – 349 (2002)
19. Nakano, K.: Optimal initializing algorithms for a reconfigurable mesh. Journal of Parallel and Distributed Computing **24**(2), 218–223 (1995)
20. Nakano, K., Olariu, S.: Energy-efficient initialization protocols for radio networks with no collision detection. In: International Conference on Parallel Processing, 2000. pp. 263–270 (2000)
21. Nakano, K., Olariu, S.: Energy-efficient initialization protocols for single-hop radio networks with no collision detection. IEEE Transactions on Parallel and Distributed Systems **11**(8), 851–863 (2000)
22. Nakano, K., Olariu, S.: Uniform leader election protocols for radio networks. IEEE Trans. on Parallel Distrib. Syst. **13**(5), 516 – 526 (2002)
23. Raab, M., Steger, A.: "balls into bins"—a simple and tight analysis. In: International Workshop on Randomization and Approximation Techniques in Computer Science. pp. 159–170. Springer (1998)
24. Ramanathan, M.K., Ferreira, R.A., Jagannathan, S., Grama, A., Szpankowski, W.: Randomized leader election. Distributed Computing **19**(5-6), 403–418 (2007)
25. Scott, A., Jeavons, P., Xu, L.: Feedback from nature: an optimal distributed algorithm for maximal independent set selection. In: Proceedings of the 2013 ACM symposium on Principles of distributed computing. pp. 147–156. ACM (2013)
26. Vlady, R.: Time-optimal and energy-efficient size approximation of radio networks. In: Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on. pp. 233–237. IEEE (2016)
27. Yao, A.C.: Probabilistic computations: Toward a unified measure of complexity (extended abstract). In: 18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977. pp. 222–227 (1977)