On the Complexity of Universal Leader Election

Shay Kutten[*]
Faculty of IE&M,
Technion, Haifa 32000.
kutten@ie.technion.ac.il

Gopal Pandurangan[† ‡]
Div. of Mathematical Sciences,
Nanyang Technological Univ.,
Singapore 637371.
gopalpandurangan@gmail.com

David Peleg[§]
Dept. of Computer Science,
The Weizmann Institute,
Rehovot, Israel.
david.peleg@weizmann.ac.il

Peter Robinson[¶]
Div. of Mathematical Sciences,
Nanyang Technological Univ.,
Singapore 637371.
peter.robinson@ntu.edu.sg

Amitabh Trehan[*]
Faculty of IE&M,
Technion, Haifa 32000.
amitabh.trehaan@gmail.com

ABSTRACT

Electing a leader is a fundamental task in distributed computing. In its *implicit* version, only the leader must know who is the elected leader. This paper focuses on studying the message and time complexity of *randomized* implicit leader election in synchronous distributed networks. Surprisingly, the most "obvious" complexity bounds have not been proven for randomized algorithms. The "obvious" lower bounds of $\Omega(m)$ messages (m is the number of edges in the network) and $\Omega(D)$ time (D is the network diameter) are non-trivial to show for randomized (Monte Carlo) algorithms. (Recent results that show that even $\Omega(n)$ (n is the number of nodes in the network) is *not* a lower bound on the messages in complete networks, make the above bounds somewhat less obvious). To the best of our knowledge, these basic lower bounds have not been established even for deterministic algorithms (except for the limited case of comparison algorithms, where it was also required that some nodes may not wake up spontaneously, and that D and n were not known).

We establish these fundamental lower bounds in this paper for the general case, even for randomized Monte Carlo algorithms. Our lower bounds are universal in the sense that they hold for all universal algorithms (such algorithms should work for all graphs), apply to every D, m, and n, and hold even if D, m, and n are known, all the nodes wake up simultaneously, and the algorithms can make any use of node's identities. To show that these bounds are tight, we present an $O(m)$ messages algorithm. An $O(D)$ time algorithm is known. A slight adaptation of our lower bound technique gives rise to an $\Omega(m)$ message lower bound for randomized broadcast algorithms.

An interesting fundamental problem is whether *both* upper bounds (messages and time) can be reached *simultaneously* in the randomized setting for all graphs. (The answer is known to be negative in the deterministic setting). We answer this problem partially by presenting a randomized algorithm that matches both complexities in some cases. This already separates (for some cases) randomized algorithms from deterministic ones. As first steps towards the general case, we present several universal leader election algorithms with bounds that trade-off messages versus time. We view our results as a step towards understanding the complexity of universal leader election in distributed networks.

Categories and Subject Descriptors

F.2.2 [**Theory of Computation**]: [Analysis of Algorithms and Problem Complexity—Nonnumerical Algorithms and Problems]

Keywords

Leader election; lower bound; distributed algorithm.

1. INTRODUCTION

Leader election is a fundamental and classical problem in distributed computing. Due to shortage of space, we rely on the reader's familiarity with its long history and many theoretical implications. Previous work is too rich to survey

[*]Research supported in part by the Israel Science Foundation and by the Technion TASP center.

[†]Research supported in part by the following grants: Nanyang Technological University grant M58110000, Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 2 grant MOE2010-T2-2-082, MOE AcRF Tier 1 grant MOE2012-T1-001-094, and a grant from the US-Israel Binational Science Foundation (BSF).

[‡]Also affiliated with the Department of Computer Science, Brown University, Box 1910, Providence, RI 02912, USA.

[§]Supported in part by the Israel Science Foundation (grant 894/09), the United States-Israel Binational Science Foundation (grant 2008348), the Israel Ministry of Science and Technology (infrastructures grant), and the Citi Foundation.

[¶]Research supported in part by the following grants: Nanyang Technological University grant M58110000, Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 2 grant MOE2010-T2-2-082.

PODC'13, July 22–24, 2013, Montréal, Québec, Canada.

here, see, e.g., [3, 14, 19, 20]. Still, let us stress that this long-studied task is relevant today more than ever, with its practical applications to the emerging area of large scale and resource-constrained networks such as peer-to-peer networks (e.g., that of Akamai [16]), ad hoc and sensor networks (e.g., [9, 21]). For example, minimizing messages and time for basic tasks such as leader election can help in minimizing energy consumption in ad hoc and sensor networks. Hence, it is desirable to achieve fast, low cost and scalable leader election. This is one of the reasons why this paper concentrates on randomized algorithms, that have been shown to reduce complexity dramatically in various contexts. (In fact, it was recently shown that the randomized message complexity of leader election in complete graphs is sublinear, $O(\sqrt{n}\log^{3/2} n)$[12].) Interestingly, although the leader election task is so well studied, some basic theoretical questions concerning its complexity have not been answered yet, especially (but not only) for the randomized case.

Informally, the leader election task requires a group of processors in a distributed network to elect a unique leader among themselves, i.e., exactly one processor must output the decision that it is the leader, say, by changing a special *status* component of its state to the value *leader*, with all the other nodes changing their *status* component to the value *non-leader*. These nodes need not be aware of the identity of the leader. This *implicit* version of leader election is rather standard (cf. [14]), and is sufficient in many applications, e.g., its original application for token generation in a token ring environment [13]. (In the *explicit* variant, every node must also know the identity of the unique leader.) This paper *focuses on implicit leader election*, although our algorithms apply to the explicit version as well.

The study of leader election algorithms is usually concerned with both message and time complexity. Both may appear to be very well understood. For example, Awerbuch [4] presented a deterministic algorithm (that takes $O(n)$ rounds and uses $O(m + n\log n)$ messages) that was claimed to be optimal both in terms of time complexity and in terms of message complexity. As demonstrated in [17], the time in Awerbuch's algorithm may be optimal only existentially, that is, only in the case that $n = O(D)$, where n is the number of nodes, and D the diameter of the graph. Moreover, these claims of optimality rely on the tacit assumption that D and m (the number of edges) are lower bounds on the time and the number of messages required for leader election, respectively. Surprisingly, even for deterministic algorithms, the only proof we are aware of for a lower bound of $\Omega(m)$ on the message complexity of leader election is for the rather limited case where (a) the algorithms are only *comparison* algorithms (that may not manipulate the actual value of node's identities, but only compare identities with each other), (b) spontaneous wakeup of the nodes is not guaranteed, and (c) network parameters (such as n) are not known to the nodes. This limited case admits a very short and elegant proof, cf. [20]. Unfortunately, that proof fails completely if one of the assumptions is removed[1]. (As a by product of our results for randomized algorithms, we get rid of all these special assumptions for deterministic algorithms too.) For the time complexity of leader election, the situation is even less stable, and the lower bound of D seems to be folklore, cf. [5, 20].)

Let us assume for a moment that the above lower bounds were indeed "obvious" (though not formally proven for the general case) for deterministic algorithms. Are they indeed that obvious for randomized ones? The work of [12] demonstrated that, for randomized algorithms, the "obvious" lower bound of $\Omega(n)$ messages for a complete graph (as well as other classes of graphs with sufficiently small mixing times such as expanders and hypercubes) does not hold. Specifically, it presented an algorithm that executes in $O(1)$ time and uses only $O(\sqrt{n}\log^{3/2} n)$ messages to elect a leader in a complete graph (and similarly for other families of high-expansion graphs). Consequently, it would appear that the obvious lower bounds on time and messages must be revisited, especially for randomized algorithms.

This paper concerns *universal* leader election algorithms, namely, algorithms that work for all graphs. The unconditional randomized lower bounds of $\Omega(m)$ messages and $\Omega(D)$ time shown in this paper (cf. Table 1) subsume the above deterministic bounds in a general way. These bounds apply to a large class of graphs for (essentially) every given m, D, and n. They hold even for non-comparison algorithms and even if nodes have knowledge of these parameters. They also hold for synchronous networks, and even if all the nodes wake up simultaneously. Finally, they hold not only for the $\mathcal{CONGEST}$ model [18] (where sending a message of $O(\log n)$ bits takes one unit of time) but also for the $\mathcal{LOCAL}$ model (where the number of bits in a message is allowed to be arbitrary).

We note that the universal lower bounds of $\Omega(m)$ and $\Omega(D)$ do not follow from currently known results. There are several known lower bounds for *deterministic* leader election algorithms in cycles (e.g., [7]) and complete graphs (e.g., [11, 2]), which also imply bounds for (deterministic) *universal* algorithms. These results alone do not, however, imply that a universal algorithm cannot do significantly better in other classes of networks. For example, the deterministic lower bound of $\Omega(n\log n)$ messages in ring graphs (cf. [7, 20]) does not imply a lower bound of $\Omega(m)$ messages (even for deterministic algorithms) in general graphs. Moreover, it does not even imply the necessity of $\Omega(n\log n)$ messages for every graph, since there exist graphs with n nodes where the number of messages required is smaller (e.g., a star graph). It may even be possible to design a universal election algorithm that will use only $O(n)$ messages when executed on a star graph.

Compared to deterministic algorithms, lower bounds (and their proofs) for randomized algorithms, particularly Monte Carlo ones, are more delicate. For example, consider the following simple algorithm (which assumes the nodes know n): "Each node elects itself as leader with probability $1/n$." The probability of this algorithm resulting in exactly one leader is $\binom{n}{1}\frac{1}{n}(1-1/n)^{n-1} \approx 1/e \approx 0.368$. Hence there exists a randomized algorithm that elects a leader in one

[1]Interestingly, even in the *explicit* deterministic variant, an "obvious" lower bound of $\Omega(m)$ messages was *not* known. Indeed, the explicit version seems to require a broadcast of the leader's name. Still, the known lower bound of $\Omega(m)$ on deterministic broadcast was shown only for the case where the nodes do not wake up simultaneously [5]. Hence in the general case, it was not known that conveying the leader's identity to every node consumes $\Omega(m)$ messages. As opposed to the number of edges, still for the explicit variant, $\Omega(n)$ (the number of *nodes*) and $\Omega(D)$ do seem to be known lower bounds on the messages and time. Nevertheless, $\Omega(D)$ time is not obvious for the implicit variant studied here.

time unit, without sending any messages, and succeeds with constant (albeit small) probability! In contrast, as proved later in the paper, if the success probability is required to be a somewhat larger constant, then the time lower bound becomes $\Omega(D)$ and the message lower bound becomes $\Omega(m)$.

To the best of our knowledge, previous work on time complexity bounds for leader election in general networks is scarce. In a recent work[8], the authors study *deterministic* algorithms for variants of leader election in *anonymous* networks called *weak* (resp., *strong*) leader election, which require the algorithm to elect a leader in the given network if possible. It is shown in [8] that $D + \lambda$ is a lower bound in this setting, where λ is a *symmetry* parameter, which is the smallest depth at which the views of the nodes become distinguishable.

Over two decades ago, the following basic open problem was raised in [17]: Is it possible to design a universal algorithm for leader election that is *simultaneously both time and message optimal*? In view of the lower bounds in this paper, this question can be reformulated as follows: Is there an $O(D)$ time and $O(m)$ messages universal leader election algorithm? The answer is negative if we restrict ourselves to deterministic algorithms, since it is known that for a cycle any $O(n)$ time deterministic algorithm requires at least $\Omega(n \log n)$ messages (even when nodes know n) [7]. However, the problem still stands for randomized algorithms. We provide a partial answer for this problem by presenting a randomized algorithm that matches both complexities for $m \geqslant n^{1+\varepsilon}$ (for any fixed constant $\varepsilon > 0$), assuming n is known. This already separates randomized algorithms from deterministic ones. Another such case (for every m) when both lower bounds can be matched is when n and D are known. As first steps for the more general case, we present several universal leader election algorithms with bounds that trade-off messages versus time. In particular, to also show that our lower bounds are tight, we present a simple deterministic algorithm that uses $O(m)$ messages. An $O(D)$ time algorithm is already known [17].

1.1 Our Results

This paper presents lower and upper bounds for universal leader election algorithms in synchronous arbitrary networks. Our results on leader election are summarized in Table 1. The formal statements of our algorithms and their full proofs are in the full paper. The tight bounds are the lower ones – Theorem 3.13 and Theorem 3.1, as demonstrated by Theorem 4.1 (and [17]). Corollary 3.12 shows that a slight adaptation of our lower bound technique implies an $\Omega(m)$ message lower bound for solving the broadcast problem. Note that Theorem 4.3.(B) presents a case where one can match *both* lower bounds simultaneously with a constant (though close to 1) probability. Corollary 4.4 shows a case where both bounds can be matched with high success probability[2] (but with a constraint on m). Corollary 4.5 demonstrates a case with probability 1, but with an extra assumption (knowledge of D). The other results in the table may be of interest by themselves. They were obtained on the way to reaching the above results, or in trying to get close to a tight upper bound for the general case.

[2]Throughout, "with high probability (w.h.p.)" means with probability at least $1 - 1/n$.

2. PRELIMINARIES

We consider a system of n nodes, represented as an undirected connected (not necessarily complete) graph $G = (V, E)$. Each node u runs an instance of a distributed algorithm and has a unique identifier $\texttt{ID}_u$ of $O(\log n)$ bits chosen by an adversary from an arbitrary set of integers Z of size n^4 (the nodes themselves may not have knowledge of n, nor of Z). The lower bounds hold even when nodes have unique identities (IDs). However, for some of the algorithms, we do not assume that the nodes have unique identities (IDs). Hence, the randomized algorithms in this paper also work for anonymous networks. To make the lower bounds more general, we assume that all nodes wake up simultaneously at the beginning of the execution.

The computation advances in synchronous rounds, where in every round, nodes can send messages, receive messages that were sent in the same round by neighbors in G, and perform some local computation. Our algorithms work in the $\mathcal{CONGEST}$ model [18], where in each round a node can send at most one message of size $O(\log n)$ bits on a single edge. In contrast, our lower bounds apply even in the $\mathcal{LOCAL}$ model [18], where there is no restriction on message size.

For randomized (Las Vegas and Monte Carlo) algorithms, we also assume that every node has access to the outcome of unbiased private coin flips. Messages are the only means of communication; in particular, nodes cannot access the coin flips of other nodes, and do not share any memory. The classical leader election literature distinguishes between the *simultaneous wakeup model* where all nodes are awake initially and start executing the algorithm simultaneously, and the *adversarial wakeup model* where the nodes are awoken at arbitrary points in time, with the restriction that nodes wake upon receiving a message and at least one node is initially awake. Our lower bounds hold even if the nodes are initially awake. In contrast, the analysis of some of the algorithms holds even for the case of adversarial wakeup.

Initially, each node is given a port numbering where each port is connected to an incident edge leading to a neighbor. However, the node has no knowledge of the neighbor at the other endpoint of edge. Recall that our lower bounds hold even if the nodes know some of the graph parameters, such as n, D, and m. Some of our algorithms work without this assumption. However, other algorithms rely on the assumption that the nodes know one or more of these parameters (cf. Table 1).

2.1 Leader Election (LE)

We now define the leader election problem formally. Every node u has a special variable $\texttt{status}_u$ that can be set to a value in $\{\perp, \text{NON-ELECTED}, \text{ELECTED}\}$; initially $\texttt{status}_u = \perp$. An *algorithm A solves leader election in T rounds* if, from round T on, *exactly one* node has its status set to ELECTED while all other nodes are in state NON-ELECTED.

We say that A is a *universal* leader election algorithm, with error probability ε, if for any choice of n and m, the probability that A succeeds is at least $1 - \varepsilon$ on any network of n nodes and m edges, and any ID assignment chosen from any integer set of large polynomial (in n) size; for our lower bounds we assume that $|Z| \geqslant n^4$. In particular, if A is a deterministic algorithm or a randomized Las Vegas algorithm, then A is universal if and only if it achieves leader election on every network under all ID assignments.

	Time	Messages	Knowledge	Success Probability
Lower Bounds:				
Theorem 3.13	–	$\Omega(m)^{\dagger}$	n, m, D	$\geqslant 53/56$
Theorem 3.1	$\Omega(D)^{*}$	–	n, m, D	$\geqslant 15/16 + \varepsilon$ ††
Randomized Algorithms:				
Theorem 4.3	$O(D)$	$O(m \min(\log f(n), D))^{\dagger, \#}$	n	$\geqslant 1 - 1/e^{\Theta(f(n))}$
Theorem 4.3.(A)	$O(D)$	$O(m \min(\log \log n, D))^{\dagger}$	n	$\geqslant 1 - n^{-1}$
Theorem 4.3.(B)	$O(D)$	$O(m)^{\dagger}$	n	$\geqslant 1 - \varepsilon$ ††
Corollary 4.4	$O(D)$	$O(m)^{\dagger}$ if $m \geqslant n^{1+\varepsilon}$ ††	n	$\geqslant 1 - n^{-1}$
Corollary 4.2	$O(D)$	$O(m \min(\log n, D))^{\$}$	–	$\geqslant 1 - n^{-1}$
Corollary 4.5	$O(D)^{\dagger}$	$O(m)^{\dagger}$	n, D	1
Theorem 4.6	$O(D \log n)^{\$}$	$O(m + n \log n)^{\$}$	n	$\geqslant 1 - n^{-1}$
Deterministic Algorithms:				
Theorem 4.7	$O(D \log n)$	$O(m \log n)$	–	
Theorem 4.1	arbitrary	$O(m)$	–	

$^{\$}$ with high probability. † in expectation. * with constant probability. $^{\#}$ for any $f(n) \in \Omega(1)$. †† for any fixed constant $\varepsilon > 0$.

Table 1: Upper and lower bounds for universal leader election algorithms.

3. RANDOMIZED LOWER BOUNDS

As mentioned earlier, to the best of our knowledge, lower bounds for randomized (especially Monte Carlo) algorithms have not been studied. Here we present two basic lower bounds that apply to randomized algorithms, including Monte Carlo algorithms with (suitably large) *constant* success probability: an $\Omega(m)$ bound on the number of messages and an $\Omega(D)$ bound on the time. Our message lower bound applies to any m and n, i.e., we show that given any n and m, there exists a graph with $\Theta(n)$ nodes and $\Theta(m)$ edges, where the lower bound holds. Our time lower bound states that for every n and D ($2 < D < n$), there exists a graph with $\Theta(n)$ nodes and $\Theta(D)$ diameter for which the time needed is $\Omega(D)$, even with constant success probability. Also, these lower bounds hold even if the nodes have knowledge of the global parameters n, D and m. Our lower bounds apply to all algorithms (and not just comparison-based ones) and hold even if the all nodes wake up simultaneously.

Our message lower bound is proved by first showing a lower bound for a related problem referred to as *"bridge crossing (BC)"*. In the bridge crossing problem, it is required that at least one message is sent across a "bridge" edge connecting two specific subgraphs. We then show conditions under which any universal leader election algorithm must solve BC. We first show a lower bound of the expected message complexity of deterministic algorithms for BC and LE, and then use Yao's lemma (cf. Lemma 3.2) to show that it applies also to the expected message complexity of randomized algorithms on the worst-case input. The proof involves constructing a suitable graph that ensures that a knowledge of n, m, D, and identity assignment will be useless to any algorithm; it also involves a counting argument to lower bound the number of messages sent.

Our $\Omega(D)$ time lower bound is proven directly for Monte Carlo algorithms by a probabilistic argument. We show that for a suitably constructed graph of a given size and diameter, any Monte Carlo algorithm that needs to succeed with a suitably large constant probability must communicate over a distance of $\Omega(D)$ edges. Otherwise, there might not be a unique leader.

3.1 Message Complexity Lower Bound

Theorem 3.1. *Let R be a universal leader election algorithm that succeeds with probability at least $1 - \beta$, for some constant $\beta \leqslant 3/56$. For every sufficiently large n and $n \leqslant m \leqslant \binom{n}{2}$, there exists a (connected) graph G of n nodes and $\Theta(m)$ edges, such that the expected number of messages used by R on G is $\Omega(m)$. The above holds even if n, m and D are known to the algorithm and all nodes wake up simultaneously.*

For simplicity, we first describe the proof assuming the nodes do not know the diameter D. At the end of the proof, we explain why this proof fails for weaker algorithms, which are guaranteed to work correctly only when the nodes know D, and then outline the modifications necessary to allow the proof to handle this harder case as well.

The proof is based on constructing a graph family referred to as *dumbbell graphs*. Given R, n and m, pick one specific 2-connected graph G_0 of n nodes and m edges, and a range $Z = [1, n^4]$ of ID's. This G_0 has many instantiations, obtained by fixing the node ID assignment and the port number mapping. An ID assignment is a function $\varphi : V(G_0) \mapsto Z$. A port mapping for node v is a mapping $P_v : [1, deg_v] \mapsto \Gamma(v)$ (namely, v's neighbors). A port mapping for the graph G_0 is $P = \langle P_{v_1}, \ldots, P_{v_n} \rangle$. Every choice of φ and P yields a concrete graph $G_{\varphi,P}$. Denote the set of id's of this graph by $ID(G_{\varphi,P}) = \{\varphi(v) \mid v \in V(G_0)\}$. Let $\mathcal{G}$ be the collection of concrete graphs $G_{\varphi,P}$ obtained from G_0. For a graph $G \in \mathcal{G}$, and an edge e of G_0, the "open graph" $G[e]$ is obtained by erasing e and leaving the two ports that were attached to it empty. Let $\mathcal{G}^{open}$ be the collection of open graphs obtained from G_0.

For two open graphs $G'[e']$ and $G''[e'']$ with disjoint sets of ID's, $ID(G'[e']) \cap ID(G''[e'']) = \emptyset$, let $Dumbbell(G'[e'], G''[e''])$ be the graph obtained by taking one copy of each of these graphs, and connecting their open ports. Hence a dumbbell graph is composed of two open graphs plus two connecting edges, referred to as *bridges*. Moreover, we say that $G'[e']$ *participates on the left* and $G''[e'']$ *participates on the right* in $Dumbbell(G'[e'], G''[e''])$. (Strictly speaking, there could be two such graphs, but let us consider only one of them.

For concreteness, if $e' = (v', w')$ and $e'' = (v'', w'')$ where $ID(v') < ID(w')$ and $ID(v'') < ID(w'')$, then the graph $Dumbbell(G'[e'], G''[e''])$ contains the bridge edges (v', v'') and (w', w''). Create a collection $\mathcal{I}$ of inputs for our problem consisting of all the dumbbell graphs

$$\mathcal{I} = \{Dumbbell(G'[e'], G''[e'']) \mid G'[e'], G''[e''] \in \mathcal{G}^{open}, \\ ID(G'[e']) \cap ID(G''[e'']) = \emptyset\}.$$

Partition the collection of inputs $\mathcal{I}$ into classes as follows: for every two graphs $G', G'' \in \mathcal{G}$, define the class $\mathcal{C}(G', G'') = \{Dumbbell(G'[e'], G''[e'']) \mid e', e'' \in E(G_0)\}$, consisting of the m^2 dumbbell graphs constructed from G' and G''. Finally, create a uniform distribution Ψ on $\mathcal{I}$.

We now give a high level overview of the main ideas of the proof of Theorem 3.1.First, we would like to prove that every deterministic algorithm D that achieves LE on every graph in collection $\mathcal{I}$, has expected message complexity $\Omega(m)$ on Ψ. Yao's minimax principle (cf. Lemma 3.2) then implies that the randomized algorithm R has expected message complexity $\Omega(m)$ on some graph G^* of $\mathcal{I}$. Since every graph in the collection $\mathcal{I}$ has $2m$ edges, this implies a lower bound of $\Omega(m)$ for R.

To prove this, we define an intermediate problem on the input collection $\mathcal{I}$, called *bridge crossing* (BC). An algorithm for this problem is required to send a message on one of the two bridge edges connecting the two open graphs (from either direction). More precisely, any algorithm solving BC is allowed to start simultaneously at all nodes, and succeeds if during its execution, a message has crossed one of the two connecting bridge edges. (Note that in our model, the nodes are unaware of their neighbors' identities, and in particular, the four nodes incident to the two bridge edges are unaware of this fact.)

LEMMA 3.2 (MINIMAX PRINCIPLE, PROP. 2.6 IN [15]). *Consider a finite collection of inputs $\mathcal{I}$ and a distribution Ψ on it. Let X be the minimum expected cost of any deterministic algorithm that succeeds on at least a $1 - 2\beta$ fraction of the graphs in the collection $\mathcal{I}$, for some positive constant β. Then $X/2$ lower bounds the expected cost of any randomized algorithm R on the worst-case graph of $\mathcal{I}$ that succeeds with probability at least $1 - \beta$.*

Basic counting yields the following:

FACT 3.3. *Let the node degrees in G_0 be $d_1, \ldots, d_n$, where $d_i = deg(v_i, G_0)$ for $1 \leqslant i \leqslant n$.*

(a) The number of different possible port assignments is $K = \prod_{i=1}^{n} d_i!$.
(b) The number of different possible ID assignments is $\binom{n^4}{n}$.
(c) The number of graphs in the collection $\mathcal{G}$ is $g = K \cdot \binom{n^4}{n}$.
(d) The number of graphs in the collection $\mathcal{G}^{open}$ is $g^{open} = g \cdot m$.
(e) The number of graphs in each class $\mathcal{C}(G', G'')$ is m^2.
(f) For every graph $G' \in \mathcal{G}$, the number of graphs $G'' \in \mathcal{G}$ ID disjoint from G' is $\tilde{g} = K \cdot \binom{n^4 - n}{n}$.
(g) The number of classes $\mathcal{C}(G', G'')$ in the input collection $\mathcal{I}$ is $\tilde{h} = g \cdot \tilde{g}$.
(h) The number of dumbbell graphs in the input collection $\mathcal{I}$ is $\tilde{d} = \tilde{h} \cdot m^2$.

Also, straightforward calculations yield the following.

LEMMA 3.4. $\tilde{g} \geqslant (1 - 1/n)g$.

LEMMA 3.5. *For every deterministic algorithm A and for every two disjoint graphs $G', G'' \in \mathcal{G}$, if A achieves BC on at least εm^2 graphs in the class $\mathcal{C}(G', G'')$, for constant $0 < \varepsilon \leqslant 1$, then the expected message complexity of A on inputs taken from $\mathcal{C}(G', G'')$ with a uniform distribution is $\varepsilon^2 m/8 = \Omega(m)$.*

PROOF. Consider two disjoint graphs $G', G'' \in \mathcal{G}$ and an algorithm A satisfying the premise of the lemma. Perform the following experiment. Run the code of algorithm A on the nodes of the $2n$-node graph G'^2 composed of two (disconnected) copies of G'. Denote this execution by $EX(G')$. (Of course, since G' is not a dumbbell graph, this is not a legal input for A, so there are no guarantees on the output or even termination of this execution.) The algorithm will send some messages, and then possibly halt. For each edge e (interpreted as a directed edge), identify the first time $t(e)$ in which a message was sent over e (in this direction) in this execution. Order the (directed) edges in increasing order of $t(e)$, getting the list $\hat{E}' = (e'_1, \ldots, e'_{k'})$. (Edges on which no messages were sent are not included in this list, so $k' \leqslant 2m$.) Run a similar experiment $EX(G'')$ on G'', getting a list $\hat{E}'' = (e''_1, \ldots, e''_{k''})$.

Now consider the m^2 executions $EX(Dumbbell(G'[e'], G''[e'']))$ of A on the dumbbell graphs in the class $\mathcal{C}(G', G'')$. In at least εm^2 of these executions, the algorithm A succeeds, so (at least) one message crosses one bridge in one direction. Without loss of generality, in at least $\varepsilon m^2/2$ of these executions, the first crossing message was sent from G' to G''.

Partition the dumbbell graphs in the class $\mathcal{C}(G', G'')$ into subclasses $C_0, C_1, \ldots, C_{k'}$, where the subclass C_i for $1 \leqslant i \leqslant k'$ contains all the dumbbell graphs $Dumbbell(G'[e'], G''[e''])$ in which $e' = e'_i$ and in execution $EX(Dumbbell(G'[e'], G''[e'']))$, the first crossing message went over the edge e'_i from $G'[e']$ to $G''[e'']$. The subclass C_0 contains all the remaining graphs of the class $\mathcal{C}(G', G'')$.

Consider an execution $EX(Dumbbell(G'[e'], G''[e']))$ in which BC was achieved, and assuming that the first crossing message was sent in round t from G' to G'', say, over port p that in the original G' was used for e'. A crucial observation is that the part of this execution restricted to $G'[e']$ is identical to the execution $EX(G')$ up to and including round t. Similarly, the part of this execution restricted to $G''[e'']$ is identical to the execution $EX(G'')$ up to and including round t. This implies that e' must occur in the list $\hat{E}'$ in some position, as e'_j. In particular, it follows that the subclass C_0 contains only dumbbell graphs in which the first crossing message went from $G''[e'']$ to $G'[e']$ plus all the dumbbell graph in which no message crossed. In addition, it follows that $|C_0| \leqslant (1 - \varepsilon)m^2/2$ and $\sum_{i=1}^{k'} |C_i| \geqslant \varepsilon m^2/2$. Moreover, in the execution $EX(Dumbbell(G'[e'], G''[e']))$, algorithm A must have sent at least one message on each of the edges e'_i for $1 \leqslant i \leqslant j$, i.e., A must have sent at least j messages.

We define $\ell_j = |C_j|$ to be the number of executions $EX(Dumbbell(G'[e'], G''[e']))$ in which the first crossing message was sent from G' to G'' over the edge e'_j. This requires, in particular, that $e' = e'_j$. As there are exactly m dumbbell graphs $Dumbbell(G'[e'_j], G''[e''])$, it follows that $\ell_j \leqslant m$. Let $B = \sum_{j=1}^{k'} \ell_j$. By assumption, $B \geqslant \varepsilon m^2/2$. Let Q be the total number of messages sent by A in all these executions. Then $Q \geqslant \sum_{j=1}^{k'} \ell_j \cdot j$. To lower bound Q, note that this last sum is minimized if the first B/m summands are $\ell_i = m$, for

$i = 1, \ldots, B/m$, and the remaining summands are 0. Hence

$$Q \geqslant \sum_{j=1}^{B/m} m{\cdot}j \geqslant \frac{m}{2}{\cdot}\frac{B}{m}{\cdot}\left(\frac{B}{m}+1\right) \geqslant B^2/(2m) \geqslant \varepsilon^2 m^3/8.$$

Therefore the expected cost incurred by A over the class $\mathcal{C}(G', G'')$ is at least $\varepsilon^2 m/8 = \Omega(m)$. $\square$

LEMMA 3.6. *Every deterministic algorithm A that achieves BC on at least $1/4$ of the dumbbell graphs in the collection $\mathcal{I}$ has expected message complexity $\Omega(m)$ on Ψ.*

PROOF. Consider an algorithm A as in the lemma. Let z denote the number of dumbbell graphs in the collection $\mathcal{I}$ on which algorithm A achieves BC. By the assumption of the lemma, $z \geqslant \tilde{d}/4$. Let X denote the set of pairs of disjoint graphs (G', G'') such that algorithm A achieves BC on at least $m^2/8$ of the m^2 dumbbell graphs $Dumbbell(G'[e'], G''[e''])$ in the class $\mathcal{C}(G', G'')$. Let Y denote the set of remaining pairs (such that A achieves BC on fewer than $m^2/8$ of the dumbbell graphs in $\mathcal{C}(G', G'')$). Let $x = |X|$ and $y = |Y|$. Note that $x + y = \tilde{h}$.

CLAIM 3.7. $x \geqslant \tilde{h}/8$.

PROOF. Observe that z, the number of dumbbell graphs in $\mathcal{I}$ on which algorithm A achieves BC, cannot exceed $xm^2 + ym^2/8$, hence

$$xm^2 + ym^2/8 \geqslant z \geqslant \tilde{d}/4 = \tilde{h}m^2/4.$$

Hence assuming, to the contrary, that $x < \tilde{h}/8$, implies that

$$\frac{\tilde{h}}{8}\cdot m^2 + \frac{7\tilde{h}}{8}\cdot\frac{m^2}{8} > xm^2 + ym^2/8 \geqslant \frac{\tilde{h}m^2}{4},$$

or $15/64 > 1/4$, contradiction. $\square$

By Lemma 3.5, for every pair of disjoint graphs $(G', G'') \in X$, the expected message complexity of A on inputs taken from $\mathcal{C}(G', G'')$ with a uniform distribution is at least $m/2^7$. Hence the total number of messages sent by the algorithm when executed over all inputs from $\mathcal{C}(G', G'')$ is at least, $(xm^2)\cdot m/2^7 + (ym^2)\cdot 0$. The first summand stands for the xm^2 graphs in the x classes of X, and the second summand stands for the graphs in the classes of Y. By Claim 3.7, this is at least $(\tilde{h}/2^3)\cdot(m^3/2^7) = \tilde{d}m/2^{10}$, hence the expected message complexity of algorithm A over all disjoint graph pairs in $\mathcal{I}$ (with a uniform distribution) is at least $m/2^{10} = \Omega(m)$.

LEMMA 3.8. *Let ε and $\delta \geqslant 1/4$ be positive constants such that $7\varepsilon + \delta \leqslant 1$. If a deterministic universal LE algorithm A solves LE on at least a $1-\varepsilon$ fraction of the input graphs in $\mathcal{I}$, then A achieves BC on at least a δ fraction of the graphs in $\mathcal{I}$.*

PROOF. Denote by $\mathcal{I}^{LE}$ (respectively, $\mathcal{I}^{BC}$) the set of input dumbbell graphs on which algorithm A achieves LE (resp., BC). Let $\mathcal{I}^* = \mathcal{I}^{LE} \setminus \mathcal{I}^{BC}$. By the assumption of the lemma, $|\mathcal{I}^{LE}| \geqslant (1-\varepsilon)\tilde{d}$. Assume, towards contradiction, that $|\mathcal{I}^{BC}| < \delta\tilde{d}$. Then

$$|\mathcal{I}^*| > (1-\varepsilon-\delta)\tilde{d}. \quad (1)$$

Let W denote the set of open graphs $G'[e']$ that participate on the left in dumbbell graphs in $\mathcal{I}^*$. Formally,

$$W = \{G'[e'] \in \mathcal{G}^{open} \mid \exists G''[e''] \text{ s.t. } (G'[e'], G''[e'']) \in \mathcal{I}^*\}.$$

Let $Z = \mathcal{G} \setminus W$. Note that $|W| + |Z| = gm$. Observe that

$$(1-\varepsilon-\delta)g\tilde{g}m^2 = (1-\varepsilon-\delta)\tilde{d} < |\mathcal{I}^*| \leqslant |W|\tilde{g}m. \quad (2)$$

The last inequality follows from the fact that we can combine $G'[e'] \in W$ to form a dumbbell with any $G''[e'']$ of the $\tilde{g}$ disjoint graphs where e'' can be any one of m edges.

OBSERVATION 3.9. $|W| > (1-\varepsilon-\delta)gm$.

COROLLARY 3.10. $|Z| < (\varepsilon+\delta)gm$.

Consider an execution of algorithm A on a dumbbell graph $(G'[e'], G''[e'']) \in \mathcal{I}^*$. Necessarily, in one of the two graphs, all nodes ended in state NON-ELECTED, and in the other graph, exactly one node ended in state ELECTED and all the others in state NON-ELECTED. Suppose all nodes in $G'[e']$ ended in state NON-ELECTED. Then for every other dumbbell graph $(G'[e'], G'''[e''']) \in \mathcal{I}^*$ or $(G'''[e'''], G'[e']) \in \mathcal{I}^*$ in which $G'[e']$ participates, the run on $G'[e']$ will behave the same as in the run on $(G'[e'], G''[e''])$, so all nodes in $G'[e']$ will end in state NON-ELECTED.

This observation implies that the open graphs in W can be partitioned into two sets:

- the set W_{NE} of graphs $G'[e'] \in W$ for which in executions on dumbbell graphs belonging to $\mathcal{I}^*$, all nodes in $G'[e']$ end in state NON-ELECTED, and
- the set W_E of graphs $G'[e'] \in W$ for which in executions on dumbbell graphs belonging to $\mathcal{I}^*$, exactly one node ends in state ELECTED and all other nodes end in state NON-ELECTED.

For every open graph $G'[e'] \in W_{NE}$, let

$$\Gamma(G'[e']) = \{G''[e''] \in \mathcal{G}^{open} \mid G'[e'] \text{ and } G''[e''] \text{ are disjoint}\}.$$

Also let

$$\begin{aligned}\Gamma_{NE}(G'[e']) &= W_{NE} \cap \Gamma(G'[e']),\\ \Gamma_E(G'[e']) &= W_E \cap \Gamma(G'[e']),\\ \Gamma_Z(G'[e']) &= Z \cap \Gamma(G'[e']).\end{aligned}$$

Denote the sizes of these sets by $\gamma(G'[e'])$, $\gamma_{NE}(G'[e'])$, $\gamma_E(G'[e'])$, and $\gamma_Z(G'[e'])$, respectively. Note that

$$\gamma_{NE}(G'[e']) + \gamma_E(G'[e']) + \gamma_Z(G'[e']) = \gamma(G'[e']) = \tilde{g}m. \quad (3)$$

We separate the analysis into two cases.

Case 1: $|W_{NE}| > |W|/2$: By Obs. 3.9, $|W_{NE}| > (1-\varepsilon-\delta)gm/2$. By the assumption of Case 1, $\gamma_E(G'[e']) \leqslant |W_E| \leqslant |W|/2 \leqslant gm/2$. By Cor. 3.10, $\gamma_Z(G'[e']) \leqslant |Z| < (\varepsilon+\delta)gm$. Combining the last two facts with Eq. (3) and Lemma 3.4, implies that

$$\begin{aligned}\gamma_{NE}(G'[e']) &= \tilde{g}m - (\gamma_E(G'[e']) + \gamma_Z(G'[e']))\\ &\geqslant (1-1/n)gm - gm/2 = (1/2-1/n)gm.\end{aligned}$$

Our key observation is that every pair of open graphs from W_{NE} forms a dumbbell graph on which algorithm A fails to solve LE, since no node will end in state ELECTED. This allows us to lower bound the number X of dumbbell graphs on which agorithm A fails to solve leader election: summing over all graphs in W_{NE}, we get that

$$\begin{aligned}X &\geqslant \sum_{G'[e']\in W_{NE}} \gamma_{NE}(G'[e']) \geqslant |W_{NE}|\cdot(1/2-1/n)gm\\ &> ((1-\varepsilon-\delta)gm/2)\cdot((1/2-1/n)gm)\\ &\geqslant (1-\varepsilon-\delta)\tilde{d}/6.\end{aligned}$$

On the other hand, recall that we have assumed that $|\mathcal{I}^{LE}| \geqslant (1-\varepsilon)\tilde{d}$, so $X \leqslant \varepsilon\tilde{d}$. It follows that $(1-\varepsilon-\delta)\tilde{d}/6 < \varepsilon\tilde{d}$. Rearranging, we get that $7\varepsilon + \delta > 1$, contradicting the assumption of the lemma.

Case 2: $|W_E| > |W|/2$: A similar contradiction is derived, based on the observation that every pair of open graphs from W_E forms a dumbbell graph on which algorithm A fails to solve LE.

This completes the proof of Lemma 3.8. □

Combining Lemmas 3.6 and 3.8 allows us to use a reduction of BC to LE to show the claimed result for a universal randomized algorithm R that achieves LE with probability at least $1-\beta$: Suppose that A is a universal deterministic leader election algorithm that achieves LE on at least a $1-2\beta$ fraction of the dumbbell graphs in $\mathcal{I}$. By Lemma 3.8, we know that A achieves BC on at least a $\delta \geqslant 1/4$ fraction of the dumbbell graphs in $\mathcal{I}$. Applying Lemma 3.6 yields that A must have an expected message complexity of $\Omega(m)$ on distribution Ψ, which shows the theorem for deterministic algorithms. By a simple application of Yao's minimax principle (cf. Lemma 3.2), it follows that the $\Omega(m)$ bound for deterministic algorithms is a lower bound for the expected message complexity of R (under the worst case input), thereby completing the proof of Theorem 3.1.

At this point, let us explain why the above proof fails for weaker algorithms, which are guaranteed to work correctly only when the nodes know D. The problem occurs in the proof of Lemma 3.5. In that proof, we run an experiment where we execute algorithm A (which is now assumed to work correctly only when the nodes are given the diameter Diam(G) as part of their inputs) on an "illegal" graph G'^2 composed of two (disconnected) copies of G'. We then argue that the execution on the dumbbell graphs serving as the "real" inputs will behave the same as on the illegal graph G'^2 (so long as there was no bridge crossing). But this claim no longer holds when the nodes get the diameter D as part of their input, since the diameter of the dumbbell graph is different from that of the illegal graph G'^2 (which is infinite), so a node v will see a different input in the two execution.

To fix this problem, a natural idea would be to "feed" the nodes participating in the experiment on the illegal graph G'^2 a "fake" input on the diameter, i.e., set the input variable $DIAM_v$ at each node v to D', where D' is the diameter of the dumbbell graph. (Again, as this input is illegal, it is not guaranteed that the algorithm will generate a meaningful output, but still, the execution will behave the same as in the "real" execution on the dumbbell graph.)

A technical difficulty that prevents us from using this idea directly is that there are many dumbbell graphs for a given pair G', G'', and they have different diameters. This means that no *single* experiment (run with a specific value of D fed to the nodes) will be similar to *all* executions on all dumbbell graphs.

Hence to overcome this difficulty, it is necessary to pick G_0 and construct the collection of input graphs for the algorithm so that no matter which graphs G' and G'' are chosen, and which edges e' and e'' are crossed-over, the diameter of the resulting dumbbell graph is always the same. The observation that assists us in achieving this property is that we are free to select the graph G_0, so long as we adhere to the requirements that its size is n nodes and $\Theta(m)$ edges. So let us pick G_0 to have the following structure. Let κ be the largest integer such that $\binom{\kappa}{2} + \kappa \leqslant m$. Let G_0^1 be the complete graph on κ nodes, with $m_1 = \binom{\kappa}{2}$ edges. Let G_0^2 be a path of $n-\kappa$ nodes, $(b_1, \ldots, b_{n-\kappa})$. Combine the two graphs into G_0 by adding κ edges connecting b_1 to every node in G_0^1. It is straightforward to verify that the resulting graph G_0 satisfies the size requirements.

Next, we modify the proof by limiting the ways in which we create open graphs from G_0. Specifically, we will consider only open graphs obtained by disconnecting an edge e' in the clique G_0^1. That is, the resulting family of open graphs will contain only graphs $(G'[e'])$ for $e' \in E(G_0^1)$.

The key observation is that no matter which two edges e' and e'' of G_0^1 we choose to disconnect in G' and G'' respectively, the resulting dumbbell graph $Dumbbell(G'[e'], G''[e''])$ will always have the same diameter, $D = 2n - 2\kappa + 1$ (which is the distance between the two endpoints $b_{n-\kappa}$ of the two graphs $G'[e']$ and $G''[e'']$).

One can verify that with this change, the rest of the proof goes through, with m_1 replacing m in the intermediate claims. In particular, the number of graphs in each class $\mathcal{C}(G', G'')$ becomes m_1^2, and so on. We end up proving that any deterministic leader election algorithm uses an average of $\Omega(m_1)$ messages on the collection $\mathcal{I}$ of inputs. Those messages are sent over the edges of the κ-clique (we cannot prove that there will be any messages sent over the edges of the paths G_0^2 in either side of the dumbbell graphs). Yet as $m_1 = \Omega(m)$, this suffices to establish the desired lower bound of $\Omega(m)$ on the expected message complexity of R on the worst case input.

3.2 A Message Lower Bound for Broadcast

We can leverage our lower bound technique to show an analogous bound for the *broadcast problem* [5], where a single node must convey a message to all other nodes. In fact, we can show a lower bound of $\Omega(m)$ messages even for the weaker *majority broadcast* problem, where the message of a single node needs to reach $> n/2$ nodes.

Consider the same collection of dumbbell graphs $\mathcal{I}$ as in Lemma 3.6. If a deterministic algorithm B successfully broadcasts in some execution on a graph $G \in \mathcal{I}$, then B also achieves bridge crossing in G. The following is immediate from Lemma 3.6:

LEMMA 3.11. *Suppose that there is an algorithm B that achieves broadcast on at least $1/4$ of the graphs in $\mathcal{I}$. Then B has expected message complexity of $\Omega(m)$ on Ψ.*

By a direct application of Lemma 3.2, we get a lower bound for randomized algorithms:

COROLLARY 3.12. *Let R' be an algorithm that successfully broadcasts a message from a source node to a majority of nodes with probability at least $1-\beta$, for some constant $\beta \leqslant 3/8$. For every sufficiently large n and $n \leqslant m \leqslant \binom{n}{2}$, there exists a (connected) graph G of n nodes and $\Theta(m)$ edges, such that the expected number of messages used by R' on G is $\Omega(m)$. The above holds even if n, m and D are known to the algorithm and all nodes wake up simultaneously.*

3.3 Time Complexity Lower Bound

THEOREM 3.13. *Consider a universal leader election algorithm R that succeeds with probability $1-\beta$, where $\beta < 1/16$ in the anonymous setting and $\beta < 1/16 - \varepsilon$, if nodes*

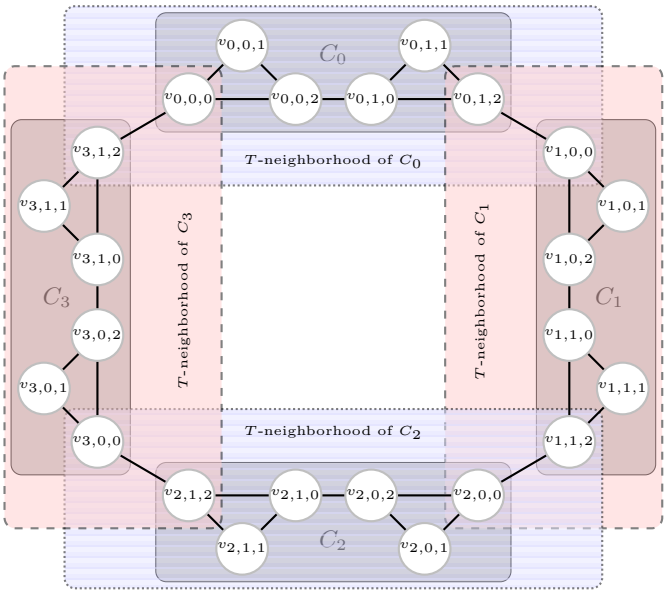

Figure 1: The Clique-Cycle Construction of Theorem 3.13 for $D' = 8$ and $n' = 24$. The blue dotted rectangles form H_Z and the red dashed rectangles correspond to $H_{Z'}$.

have unique ids, for any constant $\varepsilon > 0$. Then, for every n and every nondecreasing function $D(n)$ with $2 < D(n) < n$, there exists a graph of $n' \in \Theta(n)$ nodes and diameter $D' \in \Theta(D(n))$ where R takes $\Omega(D')$ rounds with constant probability. This is true even if all nodes know n' and D', and wake up simultaneously.

Proof. For a given n and $D(n)$, we construct the following lower bound graph G: Let $D' = 4\lceil(D(n)/4)\rceil$ and let $n' = \gamma(n)D' \in \Theta(n)$, where $\gamma(n) > 0$ is the smallest integer such that $\gamma(n)D' \geqslant n$. The graph G contains D' cliques, each of size $\gamma(n)$. Partition the cliques into 4 subgraphs $C_0, \ldots, C_3$ referred to as *arcs*, each containing $D'/4$ cliques. Let $c_{i,j}$ denote the j-th clique in the i-th arc and let $v_{i,j,k}$ be the k-th node in $c_{i,j}$, for $0 \leqslant i \leqslant 3$, $0 \leqslant j \leqslant D'/4 - 1$, and $0 \leqslant k \leqslant \gamma(n) - 1$. The graph G is a cycle C formed by the D' cliques connected the following way: For every i (adhering to the range defined above) and every $j \leqslant D'/4 - 2$, the edge $(v_{i,j,\gamma(n)-1}, v_{i,j+1,0})$ connects the clique $c_{i,j}$ to the clique $c_{i,j+1}$ within the same arc C_i. The connections between arcs C_i and $C_{i+1 \bmod 4}$ are given by the edges $(v_{i,D'/4-1,\gamma(n)-1}, v_{(i+1 \bmod 4),0,0})$. See Figure 1 for an example.

We first consider the anonymous case, where each node starts in the same initial state. Let T be the random variable denoting the running time of the assumed algorithm R and suppose that the event $T \in o(D')$ happens with probability δ. Consider the two sets of nodes formed by non-adjacent arcs $Z = (C_0, C_2)$ and $Z' = (C_1, C_3)$. Let $\phi : V(G) \to V(G)$ be a mapping such that $\phi(v_{i,j,k}) = v_{(i+1 \bmod 4),j,k}$. Let H_Z be the subgraph consisting of Z and its T-neighborhood and define $H_{Z'}$ analogously. By the definition of the adjacencies of C, it follows that the subgraph induced by $\phi(H_Z)$ is isomorphic to H'_Z.

Let *configuration* $\mathcal{C} = \langle \sigma_1, \ldots, \sigma_\ell \rangle$ denote a vector where each entry σ_i denotes a *potential local state* of a node. For a set of nodes $S = \{u_1, \ldots, u_\ell\}$, we say that S *realizes* σ *in round* r, if node u_i is in state σ_i in round r; let $E(\mathcal{C}, S, r)$ denote the event that this happens. The following claim shows that the mapping ϕ induces equi-probable realizations of any local states σ, for any pair of nodes v and $\phi(v)$. Note, however, that Claim 3.14 does *not* imply that events $E(\sigma, v, r)$ and $E(\sigma, \phi(v), r)$ are stochastically independent. The proof of the next claim follows by a symmetry argument and is deferred to the full paper:

Claim 3.14. *For any round r and any configuration $\mathcal{C}$, we have $\mathbb{P}[E(\mathcal{C}, Z, r) \mid T \in o(D')] = \mathbb{P}[E(\mathcal{C}, Z', r) \mid T \in o(D')]$.*

Note that we do not claim independence of the events $E(\mathcal{C}, Z, r)$ and $E(\mathcal{C}, Z', r)$. Let L (resp., L') be the event that one node in Z (resp., Z') becomes leader and let L_i be the event that there is one leader elected in arc C_i. It holds that $\mathbb{P}[L \mid T \in o(D')] = \mathbb{P}[L' \mid T \in o(D')]$; let q denote this probability. If the algorithm terminates in $T = o(D) = o(D')$ rounds, then there is no causal influence between C_0 and C_2, which means that $E(\mathcal{C}, C_0, r)$ and $E(\mathcal{C}, C_2, r)$ are stochastically independent (as opposed to events $E(\mathcal{C}, Z, r)$ and $E(\mathcal{C}, Z', r)$ above!) for any configuration $\mathcal{C}$ and round $r \leqslant T$. In particular, this includes all configurations that satisfy L_i. Let $p = \mathbb{P}[L_0 \mid T \in o(D')]$; again due to symmetry, we know that $\mathbb{P}[L_0 \mid T \in o(D')] = \cdots = \mathbb{P}[L_3 \mid T \in o(D')]$, and due to independence of L_0 and L_2, $q = 2p(1-p)$. Let *One* be the event that the algorithm elects 1 leader. We know that

$$\begin{aligned}\mathbb{P}\left[One \mid T \in o(D')\right] &= \frac{\mathbb{P}[One]}{\mathbb{P}[T \in o(D')]} \\ &\quad - \frac{\mathbb{P}[One \mid T \in \Omega(D')]\,\mathbb{P}[T \in \Omega(D')]}{\mathbb{P}[T \in o(D')]} \\ &\geqslant (1 - \beta - (1-\delta))/\delta = 1 - \frac{\beta}{\delta},\end{aligned}$$

and clearly

$$\begin{aligned}\mathbb{P}\left[One \mid T \in o(D')\right] &\leqslant \mathbb{P}\left[L \mid T \in o(D')\right] + \mathbb{P}\left[L' \mid T \in o(D')\right] \\ &= 2q.\end{aligned}$$

Thus $q \geqslant \frac{1}{2}(1 - \beta/\delta)$, which means that $p \geqslant \frac{1}{2}(1 - \sqrt{\beta/\delta})$. We know that $\beta = \mathbb{P}[\text{error}] \geqslant p^2$ and thus $\frac{1}{4}(1 - \sqrt{\beta/\delta})^2 \leqslant \beta$. Note that since $\beta < 1/16$, this inequality requires $\delta < 1/4$ and therefore $T \in \Omega(D')$ with constant probability. This completes the proof for the anonymous case.

Finally, for the non-anonymous case we show a reduction to the anonymous case. Now suppose that R is an algorithm designed for a setting where each node has a unique id and again assume that R takes only $o(D)$ rounds with probability $1 - \delta$, for some fixed constant δ. Consider algorithm R' that extends algorithm R by causing each node to choose an id uniformly at random from $[1, n^4]$ initially. This id is then used as input for algorithm R (instead of the id assigned in an non-anonymous network in which R is guaranteed to work). The event U, where all chosen ids are unique, occurs with probability at least $1 - n^{-2}$. By definition of R', we have $\mathbb{P}[R' \text{ succeeds} \mid U] = \mathbb{P}[R \text{ succeeds}]$ and, from the anonymous case, we know that $\mathbb{P}[R' \text{ succeeds}] \leqslant 15/16$. It follows that $\mathbb{P}[R' \text{ succeeds} \mid U] \leqslant (15/16)/\mathbb{P}[U] \leqslant 15/16(1 - n^{-2})$ for sufficiently large n. This shows that algorithm R succeeds with probability at most $15/16 + \varepsilon$, for any constant $\varepsilon > 0$. □

4. MATCHING (OR APPROACHING) THE LOWER BOUNDS

The purpose of algorithms in this paper is twofold: (1) to show the tightness of the message lower bound (the tightness of the time lower bound follows from [17]), and (2) addressing the fundamental question: can both lower bounds $O(D)$ time and $O(m)$ messages be matched or approached simultaneously? For goal (1) above, we show that there exists a deterministic universal algorithm that is optimal in the number of messages, i.e., an $O(m)$ algorithm. However, this algorithm takes arbitrary (albeit finite) time (which depends exponentially on the size of the smallest ID). This algorithm is a generalization of the one presented by Fredrickson and Lynch for rings [7].

THEOREM 4.1. *There is a deterministic leader election algorithm that uses $O(m)$ messages.*

We now briefly highlight the techniques behind the universal algorithms for goal (2), and point at their new parts (compared to previous work). We defer the full description of the algorithms to the full paper. (Our algorithms work in the $\mathcal{CONGEST}$ model).

4.1 Matching (sometimes) both lower bounds simultaneously

As mentioned earlier (cf. Section 1), only randomized algorithms have the hope of matching both lower bounds simultaneously. We start with the *least element lists (Least-El list)* algorithm of [10]. It can be used to elect a unique leader (with probability 1) in $O(D)$ time and $O(m \min(\log n, D))$ messages [10]. The main idea was to use random IDs. Each node simply floods its ID and the largest ID wins. If the IDs are chosen randomly, it can be shown that the number of messages that every node v has to forward is bounded by $O(\log n)deg(v)$, where $deg(v)$ is v's degree; this yields the desired message bound. However, in previous work, the resulting algorithm needed to know n, the number of nodes.

In the current paper, we show, first, that a standard trick can be used to get rid of this assumption, by showing that nodes can get an estimate of n (up to a constant factor) in $O(m \min(\log n, D))$ messages and $O(D)$ time: Each node u flips an unbiased coin until the outcome is heads; let X_u denote the random variable that contain the number of times that X_u is flipped. Then, nodes exchange their respective values of X_u whereas each node only forwards the highest value of X_u (once) that it has seen so far. This shows that a node can see at most D (currently) highest values. We observe that X_u is geometrically distributed and denote its global maximum by $\bar{X}$. For any u, $\mathbb{P}[X_u \geqslant 2\log_2 n] = 1/2^{2\log_2 n}$, and by taking a union bound, $\mathbb{P}\left[\bar{X} \geqslant 2\log_2 n\right] \leqslant 1/n$. Furthermore,

$$\mathbb{P}\left[\bar{X} < \log_2 n - \log_2(\log n)\right] = 1 - (1 - 1/2^{\log_2 n - \log_2(\log n)})^n \approx 1 - \exp(-n/2^{\log_2 n - \log_2(\log n)}).$$

It follows that each node forwards at most $O(\log n)$ distinct values (w.h.p.) and thus the total number of messages is $O(m \min(\log n, D))$ with high probability.

COROLLARY 4.2. *Consider a network of n nodes, m edges, and diameter D, and assume that nodes do not have knowledge of n. There is a randomized algorithm that achieves leader election with high probability, takes $O(D)$ time and has a message complexity of $O(m \min(\log n, D))$ with high probability.*

This, however, is still away from our upper bound goal to see whether it is possible to match both lower bounds. Hence, we reached another improvement to the Least-El list algorithm of [10]. That is, we show that if, instead of allowing every node to be a candidate, only $\log n$ nodes are allowed to be candidates, then the (expected) message complexity can be improved to $O(m \log\log n)$ with the same time bound. (Note that the candidates are chosen randomly and do not have any a priori knowledge of each other.) This yields a Monte-Carlo algorithm that succeeds with high probability. More generally, we show it is possible to obtain a trade-off between the success probability and the number of messages. In particular, we get an $O(m)$ messages, $O(D)$ time algorithm with large constant success probability (for any large pre-specified constant).

THEOREM 4.3. *Consider a network of n nodes, m edges, and diameter D and assume that each node knows n. Let $f(n) \leqslant n$ be any function such that $f(n) \in \Omega(1)$. There is a randomized leader election algorithm A that terminates in $O(D)$ rounds, has an expected message complexity of $O(m \cdot \min(\log f(n), D))$ and succeeds with probability $1 - 1/e^{\Theta(f(n))}$. This implies the following:*

(A) There is an algorithm that takes $O(D)$ time, has an expected message complexity of $O(m \cdot \min(\log\log n, D))$ and succeeds with high probability.

(B) For any small constant $\varepsilon > 0$, there is an algorithm that takes $O(D)$ time, sends $O(m)$ messages and succeeds with probability at least $1 - \varepsilon$.

This matches our lower bounds of time and messages for the case when the success probability is constant. However, the above improvement requires nodes to have knowledge of n.

We also managed to match both lower bounds simultaneously, for graphs that are "somewhat dense," i.e., with $m > n^{1+\varepsilon}$ for any fixed constant $\varepsilon > 0$ (known to the algorithm). This is done by combining a randomized spanner algorithm due to [6] and the above Least-El list algorithm to achieve leader election with *high* probability.

COROLLARY 4.4. *Consider a network of n nodes, m edges, and diameter D and assume that each node knows n. Let $\varepsilon > 0$ be a fixed constant. If $m \geqslant n^{1+\varepsilon}$, then there is a randomized algorithm that achieves leader election with high probability, takes $O(D)$ time and has an expected message complexity of $O(m)$.*

Finally, we show in Corollary 4.5 that if nodes have knowledge of both n and D, then one can obtain a randomized Las Vegas algorithm with expected time complexity $O(D)$ and expected message complexity $O(m)$.

COROLLARY 4.5. *Consider a network of n nodes, m edges, and diameter D, and assume that nodes have knowledge of n and D. There is a randomized algorithm that achieves leader election with probability 1, has an expected time complexity of $O(D)$ and an expected message complexity of $O(m)$.*

4.2 Approaching both lower bounds simultaneously

The Least-El list based algorithms above indeed match both lower bounds sometimes. However, at some other times

their message complexity is higher than m by a multiplicative factor that may be larger than a constant. For completeness, we also present a randomized algorithm with a better message complexity in the worst case, although at some small time penalty:

THEOREM 4.6. *Consider any network of n nodes, m edges, and diameter D, and assume that each node has knowledge of n. There is an algorithm that elects a unique leader (w.h.p.) in $O(D \log n)$ rounds and uses $O(m + n \log n)$ messages.*

This Monte-Carlo algorithm (the "clustering algorithm") is not based on the Least-El list approach, used by the other randomized algorithms above. In this algorithm (which also relies on prior knowledge of n), about $\log n$ nodes choose to be candidates and grow clusters till they essentially meet. The trick then is to first sparsify this graph and reduce the number of edges to about $\min(m, n + \log^2 n)$. However, this sparsification increases the diameter of the residual graph to $O(D \log n)$. One can then apply the Least-El list algorithm to the residual graph to obtain an overall bound of $O(D \log n)$ time and $O(m + n \log n)$ messages.

Finally, as mentioned, in the deterministic case, it is impossible to match both lower bounds simultaneously. Hence, [1] posed that goal of reaching $O(m + n \log n)$ messages and $O(D)$ simultaneously. They presented a sketch of an elegant deterministic algorithm that "grows kingdoms"; a leader candidate's kingdoms keep growing (by building BFS trees) till only one kingdom is left. Unfortunately, one can show counter examples to the hope that this algorithm doubles the diameters of winning kingdoms every round. We present a modified variant of the algorithm of [1]. The modification ensures that the (upper bound on the) number of kingdoms is reduced by a constant factor in every phase. Our algorithm requires no knowledge of n or any other parameter (it does however require unique identities, which is necessary.) This yields a $O(D \log n)$ time and $O(m \log n)$ messages deterministic algorithm. It is an open question whether the running time can be improved to $O(D)$ (with the same number of messages) in the deterministic case.

THEOREM 4.7. *Consider any network of n nodes, m edges, and diameter D. There is a deterministic algorithm that elects a unique leader in $O(D \log n)$ time while using $O(m \log n)$ messages.*

5. CONCLUSION

We studied the role played by randomization in universal leader election. Some open questions on randomized leader election are raised by our work: (1) Can we find tight (universal) upper and lower bounds for general graphs with and without knowledge of n? (2) Can we show the following conjecture: Any algorithm (even randomized Monte Carlo with high success probability) without knowledge of global parameters (e.g. n, D, m) that finishes in $O(D)$ rounds, needs at least $\Omega(m \log D)$ messages (in expectation or with high probability).

6. REFERENCES

[1] H. Abu-Amara and A. Kanevsky. On the complexities of leader election algorithms. In *ICCI*, 202–206, 1993.

[2] Y. Afek and E. Gafni. Time and message bounds for election in synchronous and asynchronous complete networks. *SIAM Journal on Computing*, 20(2):376–394, 1991.

[3] Hagit Attiya and Jennifer Welch. *Distributed Computing. (2nd ed.)*. John Wiley Interscience, 2004.

[4] B. Awerbuch. Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems. In STOC '87, pages 230–240, New York, USA, 1987. ACM.

[5] Baruch Awerbuch, Oded Goldreich, Ronen Vainish, and David Peleg. A trade-off between information and communication in broadcast protocols. *J. ACM*, 37(2):238–256, April 1990.

[6] Surender Baswana and Sandeep Sen. A simple and linear time randomized algorithm for computing sparse spanners in weighted graphs. *Random Struct. Algorithms*, 30(4):532–563, 2007.

[7] Greg N. Frederickson and Nancy A. Lynch. Electing a leader in a synchronous ring. *Journal of the ACM*, 34(1):98–115, 1987.

[8] Emanuele G. Fusco and Andrzej Pelc. Knowledge, level of symmetry, and time of leader election. In *ESA*, pages 479–490, 2012.

[9] Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. Timing-sync protocol for sensor networks. In *SenSys*, pages 138–149, 2003, New York, USA, 2003. ACM.

[10] Maleq Khan, Fabian Kuhn, Dahlia Malkhi, Gopal Pandurangan, and Kunal Talwar. Efficient distributed approximation algorithms via probabilistic tree embeddings. *Distributed Computing*, 25(3):189–205, 2012.

[11] E. Korach, S. Moran, and S. Zaks. Optimal lower bounds for some distributed algorithms for a complete network of processors. *Theoretical Computer Science*, 64(1):125 – 132, 1989.

[12] Shay Kutten, Gopal Pandurangan, David Peleg, Peter Robinson, and Amitabh Trehan. Sublinear bounds for randomized leader election. In *ICDCN'13*, pages 348–362, 2013.

[13] Gérard Le Lann. Distributed systems - towards a formal approach. In *IFIP Congress*, p. 155–160, 1977.

[14] Nancy Lynch. *Distributed Algorithms*. Morgan Kaufman Publishers, Inc., San Francisco, USA, 1996.

[15] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge Univ. Press, 1995.

[16] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. The Akamai network: a platform for high-performance internet applications. *SIGOPS Oper. Syst. Rev.*, 44(3):2–19, August 2010.

[17] David Peleg. Time-optimal leader election in general networks. *Journal of Parallel and Distributed Computing*, 8(1):96 – 99, 1990.

[18] David Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM, Philadelphia, 2000.

[19] Nicola Santoro. *Design and Analysis of Distributed Algorithms (Wiley Series on Parallel and Distributed Computing)*. Wiley-Interscience, 2006.

[20] Gerard Tel. *Introduction to distributed algorithms*. Cambridge University Press, New York, USA, 1994.

[21] Yong Yao and Johannes Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Rec.*, 31(3):9–18, 2002.