Leader Election in a Smartphone Peer-to-Peer Network

Calvin Newport
Georgetown University
Washington, D.C.
cnewport@cs.georgetown.edu

***Abstract*—In this paper, we study the fundamental problem of leader election in the *mobile telephone model*: a recently introduced variation of the classical *telephone model* modified to better describe the local peer-to-peer communication services implemented in many popular smartphone operating systems. In more detail, the mobile telephone model differs from the classical telephone model in three ways: (1) each device can participate in at most one connection per round; (2) the network topology can undergo a parameterized rate of change; and (3) devices can advertise a parameterized number of bits to their neighbors in each round before connection attempts are initiated. We begin by describing and analyzing a new leader election algorithm in this model that works under the harshest possible parameter assumptions: maximum rate of topology changes and no advertising bits. We then apply this result to resolve an open question from [1] on the efficiency of PUSH-PULL rumor spreading under these conditions. We then turn our attention to the slightly easier case where devices can advertise a single bit in each round. We demonstrate a large gap in time complexity between these zero bit and one bit cases. In more detail, we describe and analyze a new algorithm that solves leader election with a time complexity that includes the parameter bounding topology changes. For all values of this parameter, this algorithm is faster than the previous result, with a gap that grows quickly as the parameter increases (indicating lower rates of change). We conclude by describing and analyzing a modified version of this algorithm that does not require the assumption that all devices start during the same round. This new version has a similar time complexity (the rounds required differ only by a polylogarithmic factor), but now requires slightly larger advertisement tags.**

I. Introduction

This paper describes and analyzes new leader election algorithms in the *mobile telephone model*: an abstraction that captures the local peer-to-peer communication capabilities of existing commodity smartphone operating systems (e.g., as implemented by services such as Bluetooth LE [2], Wi-Fi Direct [3], and Apple's Multipeer Connectivity framework [4]). The growing ubiquity of smartphones, combined with the rapid improvement of operating system support for direct wireless communication between nearby devices, creates a compelling opportunity for the emergence of easy to deploy and widely used smartphone peer-to-peer applications.

There are several standard use cases for such applications. For example, in developing countries cellular data minutes are often bought in small blocks and carefully conserved by users—generating an interest in networking operations that can avoid infrastructure. In addition, smartphone peer-to-peer networks can be used to bypass censorship in countries where infrastructure networks are monitored (c.f., the use of peer-to-peer smartphone chat among protestors in Hong Kong [5]), and bring connectivity to settings such as disaster zones, festivals, or wilderness where traditional cellular and WiFi coverage is compromised, overwhelmed, or non-existent.

Ultimately, however, it is likely unnecessary for computer scientists to figure out *in advance* the killer app for this massive platform. If we can develop network algorithms and tools that simplify the design of useful distributed systems on top of local peer-to-peer connections, the most compelling use cases will emerge naturally from the vast competitive marketplace of smartphone application developers. With this in mind, this paper focuses on describing and analyzing new provably correct and efficient leader election algorithms that can be implemented and executed on top of existing smartphone peer-to-peer services. These algorithms provide a key primitive that supports the development of more sophisticated distributed systems by simplifying tasks such as event ordering, agreement, and synchronization.

The Mobile Telephone Model: The mobile telephone model (introduced in our recent study of rumor spreading [1]) is a variation of the classical telephone model modified to better describe smartphone peer-to-peer communication. Its details are inspired, in particular, by the current capabilities of Apple's Multipeer Connectivity framework—a peer-to-peer communication service available in every iOS version since iOS 7 (and seen as one of the more flexible smartphone peer-to-peer options at the moment). This service leverages both the 802.11 and Bluetooth radios on the smartphone to implement discovery and direct wireless connections between nearby devices. Like most smartphone peer-to-peer services, the Multipeer Connectivity framework allows devices to advertise a *service* and *scan* for the services advertised by nearby devices. A device can then attempt to form a reliable unicast *connection* with a nearby device discovered in this scan. A key limitation is that each device can only support a small number of concurrent connections.

The mobile telephone model captures these capabilities by assuming that in each round there is a graph describ-

ing the underlying network topology. At the beginning of each round, each device learns its current neighborhood (e.g., implementing a scan) and can attempt to initiate a connection with a neighbor. If two devices connect they can then perform a bounded amount of communication to conclude the round. Each device can only participate in one connection per round.

We parameterize the model with a *tag length* $b \geq 0$. At the beginning of each round, each device can choose a *tag* consisting of b bits to advertise to its neighbors. When a device scans its neighborhood it learns both the ids of its neighbors and their chosen tags. (This capability is motivated by the ability of devices to choose and change their service advertisement labels in the Multipeer Connectivity framework.) A key question in the mobile telephone model is the power of such advertisements. In our study of rumor spreading [1], we found a big complexity gap between $b = 0$ and $b = 1$, but little additional improvement for larger tags. As described below, we find a similar gap in our study of leader election.

We also parameterize our model with a *stability factor* $\tau \geq 0$. The underlying topology graph must be stable for at least τ rounds between changes. Notice, for $\tau = 1$, the network topology can change arbitrarily in every round. By contrast, for $\tau = \infty$, the network topology never changes. The need to model topology changes is important when studying smartphone peer-to-peer networks as the underlying devices are inherently mobile. In this paper, we study leader election algorithms that require no advance knowledge of τ, and can gracefully adapt to whatever level of stability they encounter in an execution.

Our Results: We study the leader election problem in the mobile telephone model. This problem assumes each device starts with a unique id (UID) and maintains a local $leader$ variable that contains an id. The problem requires the system to stabilize to a point where all devices have $leader$ set to the same UID from the network. We study how many rounds are required to reach this stabilization point with high probability in the network size. An issue with worst case analysis in this setting is that some network topologies are inherently slow (e.g., a line) while others enable the possibility of much more efficient communication (e.g., a clique). Accordingly, we include the network topology's *vertex expansion*, denoted α, (a value which ranges from a constant, indicating lots of connectivity, to something close to $1/n$, indicating very little connectivity) in our time complexity results (see Section II for a formal definition).

We begin in Section VI by studying the difficult case where $b = 0$ and $\tau = 1$; i.e., devices cannot advertise any extra information and the topology can change arbitrarily in every round. In this setting, we describe and analyze an algorithm called *blind gossip leader election* that implements the following obvious strategy: flip a coin to decide whether to receive or initiate connections; if you decide the latter choose a neighbor at random to send a connection request; if you decide the former, randomly select an incoming request (if any) to accept and use the connection to trade the smallest UIDs both devices have seen so far.

We show this algorithm stabilizes to a single leader in $O\big((1/\alpha)\Delta^2 \log^2 n\big)$ rounds, where Δ is the maximum neighborhood size and n is the network size. We then show our analysis near tight by describing a stable network in which $\Omega\big(\Delta^2/\sqrt{\alpha}\big)$ rounds are necessary for this algorithm. Finally, we leverage the new analysis techniques introduced in this section to answer an open question from [1] about rumor spreading in the mobile telephone model. In particular, we proved in [1] that efficient PUSH-PULL rumor spreading (with respect to Δ) is impossible in this model with $b = 0$, but we stopped short of proving a precise upper bound. Our analysis from Section VI allows us to resolve this question.

Next, in Section VII, we turn our attention to the slightly easier case where $b = 1$. In this setting, we describe and analyze an algorithm called *bit convergence leader election*. This algorithm has devices partition rounds into *groups* corresponding to the bits in the id of their current candidate for the leader. During a group mapped to a given bit position i, devices leverage their 1-bit tags to advertise the value in bit i in the id of their current leader candidate. Devices with a 0 in position i use these advertisements to attempt to connect with devices with a 1 in this position to send them a potentially smaller id. This task is complicated by the fact that each device can only accept a single connection per round and the graph can change every τ rounds. The devices must therefore attempt, in a fully distributed manner, to approximate a maximum matching in each group to maximize the number of unique connections between 0-bit and 1-bit devices for the current position i: a goal that becomes particularly difficult for small τ.

We show this algorithm stabilizes to a single leader in $O\big((1/\alpha)\Delta^{1/\hat{\tau}}\hat{\tau}\log^5 n\big)$ rounds, where $\hat{\tau} = \min\{\tau, \log\Delta\}$. Notice, as τ grows from 1 to $\log\Delta$, the time complexity advantage of this algorithm versus the blind gossip algorithm grows from a factor of Δ to a factor of Δ^2 (ignoring the log terms). For $\tau \geq \log\Delta$ and $\alpha = O(1)$ (e.g., a reasonably stable and well-connected network), the bit convergence algorithm requires time only polylogarithmic in the network size to stabilize.

A shortcoming of our bit convergence algorithm is that it assumes all devices activate during the same round, which is useful to the algorithm as it allows it to assume synchronized round counters. We overcome this issue in Section VIII, where we describe and analyze a variation of this algorithm that does not require devices to activate during the same round. This new variation has a similar time complexity (it is slower by only a $\log^3 n$ factor). It now also requires that $b = \log\log n + O(1)$, which is slightly larger than the $b = 1$ required by the original bit convergence algorithm. This new algorithm features characteristics of *self-stabilization* in that

if you connect isolated network components that have been running the algorithm for arbitrary durations, the combined network will still stabilize to a single leader in the same stabilization time cited above.

Our algorithm analyses build on a key graph theoretic result from [1] in which we proved a strong connection between a graph's vertex expansion and the amount of information that can concurrently travel across an arbitrary cut. The bit convergence algorithm variations also leverage a key theorem from [1] that can be interpreted as bounding the approximation ratio of random maximum matching strategies. See Section V for a detailed treatment of these useful results.

Related Work: The mobile telephone model studied in this work was introduced by Ghaffari and Newport [1]. It is a variation of the classical telephone model first introduced by Frieze and Grimmett [6]. The mobile model differs from the classical model in two ways: (1) the classical model implicitly fixes $b = 0$ and (typically) $\tau = \infty$; and (2) the classical model allows a computational process (called a *node* in the following) to accept an unbounded number of incoming connections (a property that is crucial to most analyses in the classical model). Though we discuss these differences in more detail in [1], we note that this second difference is the main reason why the classical model is not suitable for describing smartphone peer-to-peer networks—a setting in which the number of concurrent connections are typically bounded for practical reasons.

A fundamental problem in telephone-style models is *rumor spreading*: a rumor must spread from a single source to the whole network. Early studies of this problem in these models focused on restricted network topologies such as cliques (e.g., [7]), where epidemic-style spreading enables fast termination for simple random spreading strategies. In the past half-decade, attention has turned to studying rumor spreading with respect to spectral properties of the network topology graph, such as graph conductance (e.g., [8]) and, more relevant to this work, vertex expansion (e.g., [9], [10], [11], [12]). Recent work by Daum et al. [13] emphasized the well-known shortcoming of the telephone model mentioned above: it allows a single node to accept an unlimited number of incoming connections. They prove that if you instead restrict nodes to a singe connection per round, there are network topologies in which the optimal strategies from the classical model perform much worse. Our recent work [1] picks up where [13] leaves off by proving: (1) efficient rumor spreading with respect to conductance is impossible in the mobile telephone model, while efficient rumor spreading with respect to vertex expansion is possible;[1] (2) the well-studied PUSH-PULL rumor spreading strategy from the classical model cannot guarantee to be efficient with respect to α for $b = 0$; (3) a variation of PUSH-PULL called PPUSH *is* efficient with respect to α with $b = 1$, and for $\tau \geq \log \Delta$, where Δ indicates maximum degree, it *matches* the performance of PUSH-PULL in the easier classical model (within log factors).

[1]By "possible" and "impossible" we are describing the performance of an offline optimal algorithm.

Information dissemination in a key subproblem in solving leader election. Our algorithms from Section VII and VIII, which tackle the case where $b > 0$, deploy a modified version of the PPUSH rumor spreading strategy from [1] as a subroutine. Accordingly, we borrow two useful results from [1] to aid our analysis (see Section V). We emphasize that our bit convergence leader election algorithm terminates only a $\log n$ factor slower than PPUSH rumor spreading algorithm from [1] in a network of size n, even though it tackles a much more complicated task. In more detail, the PPUSH algorithm, which assumes $b = 1$, has each node use its single advertising bit to indicate whether or not it knows the single rumor in the system—this allows nodes that know the rumor to focus their connection attempts on nodes that do not. When using PPUSH to propagate UIDs for leader election, however, there is no such simple binary status to advertise. In our algorithms, all nodes, in all rounds, maintain a current candidate for leadership, so this single bit must be somehow leveraged to indicate whether a given neighbor's candidate is better or worse. This task is further complicated by a network topology that might change frequently—preventing nodes from transmitting more complicated information to their neighborhood, bit by bit, over many rounds. Our bit convergence algorithm uses this single bit in clever ways to guarantee progress is still made.

Finally, we note that leader election is well studied in many classical distributed computing models under various constraints; c.f., [14], [15], [16], [17], [18]. This problem has also been studied in models with changing network topologies; c.f., [19], [20], [21]. Perhaps most relevant to our work is the leader election algorithm from [20], which deterministically solves leader election in $O(n^2)$ rounds in a network topology that can change in every round, but which allows nodes to reliably broadcast $O(1)$ UIDs to all of their neighbors in each round. Our bit convergence algorithm solves leader election in a comparable $O(n\Delta\text{polylog}(n))$ rounds under this amount of mobility, and the worst case vertex expansion. We note, however, that in our model you can transfer a UID to at most one neighbor per round, and the n term corresponds to worst-case vertex expansion—this term can decrease to a constant in a well-connected graph.

II. Preliminaries

In this section we describe some useful notation, definitions, assumptions and probability facts.

Graph Notation: In this paper, we model network topologies with connected undirected graphs. Here we describe useful notation regarding such graphs. In particular, fix some undirected and connected graph $G = (V, E)$, defined over a

non-empty node set of V. For each $u \in V$, we use $N(u)$ to describe u's neighbors and $N^+(u)$ to describe $N(u) \cup \{u\}$. We define $\Delta = \max_{u \in V}\{|N(u)|\}$ and for each node $u \in V$, fix $d(u) = |N(u)|$. To simplify notation in our algorithm analyses, we assume Δ is a power of 2 (i.e., $\log \Delta$ is a whole number).

Vertex Expansion: For a given $S \subseteq V$, we define the *boundary* of S, indicated ∂S, as follows: $\partial S = \{v \in V \setminus S : N(v) \cap S \neq \emptyset\}$: that is, ∂S is the set of nodes not in S that are directly connected to S by an edge in E. We define $\alpha(S) = |\partial S|/|S|$. As in [12], [1], we define the *vertex expansion* α of a given graph $G = (V, E)$ as follows:

$$\alpha = \min_{S \subset V, 0 < |S| \leq n/2} \alpha(S).$$

Notice that despite the possibility of $\alpha(S) > 1$ for some S, we always have $\alpha \leq 1$.

Dynamic Graphs: Our model defined below sometimes describes the network topology with a *dynamic graph* which can change from round to round. Formally, a dynamic graph $\mathcal{G}$ is a sequence of static graphs, $G_1 = (V, E_1), G_2 = (V, E_2), ...$ When using a dynamic graph $\mathcal{G}$ to describe a network topology, we assume the r^{th} graph in the sequence describes the topology during round r. We define the vertex expansion α of a dynamic graph $\mathcal{G}$ to be the minimum vertex expansion over all of $\mathcal{G}$'s constituent static graphs. Similarly, we define the maximum degree Δ of a dynamic graph to be the maximum degree over all its static graphs.

Probability Preliminaries: Finally, we state a pair of well-known inequalities that will prove useful in our analyses:

Fact II.1. *For $p \in [0, 1]$, we have $(1 - p) \leq e^{-p}$ and $(1 + p) \geq 2^p$.*

III. The Mobile Telephone Model

We describe a smartphone peer-to-peer network using a variation of the classical telephone model called the *mobile telephone model*. In more detail, we describe a peer-to-peer network topology in each round as an undirected connected graph $G = (V, E)$. We assume a computational process (called a *node* in the following) is assigned to each vertex in V, and use $n = |V|$ to indicate the network size. Time proceeds in synchronized rounds. At the beginning of each round, we assume each node u learns its neighbor set $N(u)$. Node u can then select at most one node from $N(u)$ and send a connection proposal. A node that sends a proposal cannot also receive a proposal. However, if a node v does not send a proposal, and at least one neighbor sends a proposal to v, then v can accept an incoming proposal. There are different ways to model how v selects a proposal to accept. In this paper, for simplicity, we assume v accepts an incoming proposal selected with uniform randomness from the incoming proposals. If node v accepts a proposal from node u, the two nodes are *connected* and can perform a bounded amount of interactive communication during the round. We leave the specific bound on communication per connection as a problem parameter.

We parameterize the mobile telephone model with two integers, a *tag length* $b \geq 0$ and a *stability factor* $\tau \geq 1$. If $b > 0$, then we allow each node to select a *tag* containing b bits to advertise at the beginning of each round. That is, if node u chooses tag b_u at the beginning of a round, all neighbors of u learn b_u before making their connection decisions in this round. A node can change its tag from round to round.

We also allow for the possibility of the network topology changing, which we formalize by describing the network topology with a dynamic graph $\mathcal{G}$. We bound the allowable changes in $\mathcal{G}$ with a the stability factor τ. For a given τ, $\mathcal{G}$ must satisfy the property that at least τ rounds must pass between any changes to the graph topology. For $\tau = 1$, the graph can change arbitrarily in every round. We use the convention of stating $\tau = \infty$ to indicate the graph never changes.

IV. The Leader Election Problem

The leader election problem assumes each node begins the execution provided with a unique id (UID). We also assume each node is provided with a polynomial upper bound N on the network size n. To keep our solutions as general as possible, we treat the UID as comparable black boxes that can only be communicated between nodes through connections. In addition, we assume that a pair of connected nodes can exchange at most $O(1)$ UIDs and $O(\text{polylog}(N))$ additional bits during one round of connection.

Each node must maintain a $leader$ variable that points to a UID. These variables are initialized to each node's own UID. As a node learns other UIDs through peer-to-peer connections it can update the variable. The goal of the leader election problem is for all nodes to stabilize to a state where every $leader$ variable points to the same UID. Formally, we say the system is *stabilized* by round r, if there is some UID x such that for every round $r' \geq r$, every node $u \in V$ ends the round with $leader = x$. We say a distributed algorithm *solves the leader election problem*, if it guarantees with probability 1 that the system will eventually stabilize. We say an algorithm *solves the leader election problem in $f(n, \alpha, b, \tau)$ rounds*, if with probability at least $1 - 1/n$, the system stabilizes by round $f(n, \alpha, b, \tau)$ when executed in a network with size n, vertex expansion α, tag length b, and stability factor τ.

V. Useful Existing Results Concerning Rumor Spreading

The *rumor spreading* problem attempts to spread a single piece of information (the *rumor*) from a subset of nodes to all nodes in a network. In [1], we studied the time complexity of simple rumor spreading strategies in the mobile telephone

model. Here we replicate a pair of results from this existing study that will prove useful in our analyses of leader election algorithms in this paper.

A Formal Connection Between Expansion and Maximum Matchings: We begin by connecting graph expansion to the size of maximum matchings across network cuts. For a given graph $G = (V, E)$ and node subset $S \subset V$, we define $B(S)$ to be the bipartite graph with bipartitions $(S, V \setminus S)$ and the edge set $E_S = \{(u, v) : (u, v) \in E, u \in S, \text{ and } v \in V \setminus S\}$. Recall that the *edge independence number* of a graph H, denoted $\nu(H)$, describes the size of a maximum matching on H. For a given S, therefore, $\nu(B(S))$ describes the maximum number of concurrent connections that a network can support in the mobile telephone model between nodes in S and nodes outside of S. This property follows from the restriction in this model that each node can participate in at most one connection per round. This property is *not* true of the classical telephone model in which a given node can participate in multiple connections per round.

We now replicate an important (and non-obvious) result from our earlier investigation of the mobile telephone model. This lemma connects edge independence over cuts (the real bound of concurrent information flow) to a graph's vertex expansion (the property we use to describe our graph's connectivity):

Lemma V.1 (from [1])**.** *Fix a graph $G = (V, E)$ with $|V| = n$ with vertex expansion α. Let $\gamma = \min_{S \subset V, |S| \leq n/2} \{\nu(B(S))/|S|\}$. It follows that $\gamma \geq \alpha/4$.*

The Performance of PPUSH Rumor Spreading: We now replicate a result regarding the short term performance of a straightforward rumor spreading strategy in the mobile telephone model. For the setting where $b = 1$, an obvious approach to rumor spreading is to deploy the *productive PUSH* (PPUSH) algorithm, which works as follows: At the beginning of each round, if you know the rumor advertise tag 0, otherwise advertise tag 1. If you advertise 1, you will only receive connection proposals in this round. If you advertise tag 0, you will choose a neighbor advertising 1 (if any) uniformly at random to send a connection proposal. If a 0 connects with a 1 then the former sends the rumor to the latter.

At the beginning of any given round, we can partition the nodes into those that are *informed* (know the rumor) and those that are *uninformed* (do not know the rumor). There is a matching of some size m across this cut. This value m represent the maximum number of nodes that can become newly informed in a single round. (As noted above in Lemma V.1, this matching has a size proportional to the connectivity across the cut indicated by the graph's vertex expansion.) The following theorem from [1] bounds how well PPUSH can approximate m successful connections across the cut for a given number of stable rounds. The proof of this theorem analyzes PPUSH as a random matching strategy:

Theorem V.2 (from [1])**.** *Fix a bipartite graph G with bipartitions L and R, such $|R| \geq |L| = m$ and G has a matching of size m. Assume G is a subgraph of some (potentially) larger network G', and all uninformed neighbors in G' of nodes in L are also in R. Fix an integer r, $1 \leq r \leq \log \Delta$, where Δ is the maximum degree of G. Consider an r round execution of PPUSH in G' in which the nodes in L start with the rumor and the nodes in R do not. The exists a constant probability p and constant $c \geq 1$, such that with probability p: at least $m/f(r)$ nodes in R learn the rumor, where $f(r) = \Delta^{1/r} \cdot c \cdot r \cdot \log n$.*

VI. Leader Election with $b = 0$ and $\tau \geq 1$

We begin by considering the leader election problem in the difficult case where $b = 0$. We analyze a straightforward gossip-style strategy and prove it converges in $O((1/\alpha)\Delta^2 \log^2 n)$ rounds, even with $\tau = 1$ (i.e., the maximum possible amount of topology change). We show that the analysis is not far from optimal in the sense that there exists stable networks in which this algorithm requires at least $\Omega(\Delta^2/\sqrt{\alpha})$ rounds. We then leverage the new analysis techniques introduced here to answer an important open question from our previous study of rumor spreading [1]. We begin by describing our algorithm.

Blind Gossip Leader Election Algorithm: For each node u, let I_u describe u's UID. For each round $r \geq 0$, let $\hat{I}_u(r)$ be the smallest UID u has received by the end of round r (including its own). By definition, $\hat{I}_u(0) = I_u$. For each round $r \geq 1$, and each node u, node u flips a fair coin to determine whether to send or receive connection proposals. If u decides to send, it selects a neighbor uniformly from its neighborhood in round r. If two nodes u and v connect, they send each other $\hat{I}_u(r-1)$ and $\hat{I}_v(r-1)$, respectively. Node u sets both $I_u(r)$ and $leader$ to $\min\{\hat{I}_u(r-1), \hat{I}_v(r-1)\}$ to conclude the round.

Analysis: Our goal is to prove the following performance bound on this blind gossip strategy:

Theorem VI.1. *The blind gossip leader election algorithm solves the leader election problem in $O((1/\alpha)\Delta^2 \log^2 n)$ rounds when executed in the mobile telephone model with maximum degree Δ, vertex expansion α, stability factor $\tau \geq 1$ and tag length $b = 0$.*

The key to this analysis is bounding the time for the smallest UID in the network (call this $\hat{I}$) to spread to all nodes—at which point the network is stabilized. The intuition behind the Δ^2 term is the observation that a successful connection between a node u with $\hat{I}$ and a neighboring node v that needs to earn $\hat{I}$ requires two events to occur: (1) u decides to send a connection proposal and selects v; (2) v decides to receive connection proposals, and among the

incoming proposals, it accepts the proposal from u. In the worst-case, this probability can be $\approx \Delta^{-2}$. The $(1/\alpha)\log^2 n$ term, roughly speaking, captures the time required for these successful connections to spread $\hat{I}$ to the entire network. If $\alpha = O(1)$, for example, then the network is very well connected and an epidemic style spread can stabilize the network in only polylogarithmic rounds. If $\alpha = O(1/n)$, on the other hand, then the network is not well connected and it will take a long time for the spread of $\hat{I}$ to complete.

Two issues complicate the formalization of this intuition. The first is the changing network topology. In each round, the set of potentially useful edges can change and the definition of useful itself can change depending on the behavior in previous rounds. The second issue is probabilistic dependencies. In a given round, it is straightforward to calculate the probability that a given edge connects, but there are potential dependencies between nearby edges with respect to these probabilities.

To tame these dependency issues, we define a more pessimistic event that is sufficient (but not necessary) for a connection between a pair of neighboring nodes in a given round. Before we provide this definition we need to specify, without loss of generality, a technical detail about how our algorithm makes its random choices. In more detail, assume some node u decides to receive connection proposals in a given round. According to our model (see Section III), u will choose an incoming proposal (if any come in) with uniform randomness. Here we specify exactly how it makes this uniform choice. In particular, we assume that u first generates a random permutation of its neighbors for the round. It then receives incoming proposals and selects the proposal highest ranked in its permutation. Below we call this u's ***selection permutation*** With this detailed specified, we give the following definition:

Definition VI.2. *Fix some round $r \geq 1$. Let $\{u, v\}$ be an edge in the network topology graph for round r. Let $e = (u, v)$ be an ordered version of this pair. We say ordered edge e is* good *in round r if and only if the following events occur in this round:*

- *u decides to send connection proposals;*
- *v decides to receive connection proposals;*
- *u chooses v as the neighbor to send a proposal to; and*
- *v's selection permutation has u ranked first*

Let $X_e(r)$ be the random indicator variable that evaluates to 1 if e is good in round r, and otherwise evaluates to 0.

Notice that if an edge e is good then u and v will definitely connect. There are cases, however, where an edge is not good and u and v still connect. This condition is therefore sufficient but not necessary for a given connection. Crucially, for two edges e and e' with no nodes in common, $X_e(r)$ and $X_{e'}(r)$ are independent as the definition of good is based on events that depend only on the graph topology for the round and local independent coin flips made by individual nodes. Similarly, for any e and e' (not necessarily disjoint) and rounds $r' > r$, $X_e(r)$ and $X_{e'}(r')$ are also independent. Finally, it is straightforward to verify that for any such e and r, $\Pr(X_e(r) = 1) \geq 1/(4\Delta^2)$.

We now establish a useful graph property that follows from a direct application of Lemma V.1 from Section V. The lemma statement, as well as some of the arguments that follow, leverage the definition $B(Q)$ (the bipartite subgraph with bipartitions Q and $V \setminus Q$) also defined in Section V.

Lemma VI.3. *Fix some $Q \subset V$ such that $|Q| \leq n/2$. Fix some round $r \geq 1$. Let M be a maximum matching on $B(Q)$ defined with respect to the topology graph for round r. It follows that $|M| \geq |Q| \cdot (\alpha/4)$.*

We now prove the core technical lemmas of this analysis. In the following, let $\hat{I}$ be the smallest UID in the network, and for each round $r \geq 1$, let $S_r = \{u \mid \hat{I}_u(r-1) = \hat{I}\}$ to be the set of nodes that have adopted $\hat{I}$ by the start of round r. We will prove that with high probability S_r grows by a factor of $\approx (1+\alpha)$ in any interval of length $\Theta(\Delta^2 \log n)$ rounds. The below proofs leverage the definition of *good* from Definition VI.2 as well as Lemma VI.3 from above.

Lemma VI.4. *Fix some round $r \geq 1$ such that $|S_r| \leq n/2$. There exists a constant $c \geq 1$ such that with high probability: $|S_{r'}| \geq (1 + \frac{\alpha}{4})|S_r|$, where $r' = r + c \cdot \Delta^2 \cdot \log n$.*

Proof: Fix some r and S_r as specified by the lemma statement. Let $k = |S_r| \cdot (\alpha/4)$. Fix $t = c \cdot \Delta^2 \cdot \log n$, for a constant $c \geq 1$ that we will fix later in this proof. By Lemma VI.3 we know that in every round $r' \in R = \{r, r+1, ..., r+t-1\}$, there is a matching $M_{r'}$ of size at least k in $B(S_r)$ defined with respect to the topology graph for r' (that is, there is a matching of size k from nodes in S_r to nodes not in S_r in this round). We define a set Z of *edge/round* pairs based on these matchings as follows:

$$Z = \{((u, v), r') \mid r' \in R, \{u, v\} \in M_{r'}, u \in S_r\}.$$

In the following, for each edge/round pair $p = ((u, v), r') \in Z$, we define the notation $p.first = u$, $p.second = v$, $p.edge = (u, v)$ and $p.round = r'$. We say a given edge/round pair p is *good* if ordered edge $p.edge$ is good in round $p.round$ (see Definition VI.2). A good edge/round pair from Z indicates that a node in S_r connected to a node not in S_r—allowing S_r to grow.

We must show enough edge/round pairs are good to ensure S_r grows enough to satisfy the lemma statement in interval R. We are helped in these efforts by our careful definition of good and Z, which ensure that for $p, p' \in Z$ where $p \neq p'$, whether p is good is independent of whether p' is good (formally, $X_{p.edge}(p.round)$ and $X_{p'.edge}(p'.round)$ are independent). We can therefore calculate a lower bound

on the expected number of good edge/round pairs in Z, and then use their independence to concentrate on this mean.

Unfortunately for the cause of simplicity, bounding the number of good edge/round pairs in Z is not sufficient as many such pairs might be redundant. In particular, if two such pairs p and p' are defined such that $p.second = p'.second$, then combined they only grow S_r by one more node. To satisfy our lemma, therefore, we must take more care in organizing Z.

To do so, we start by partitioning Z into equivalence classes based on the edge endpoints not in S_r. In particular, let $Y = \{p.second \mid p \in Z\}$ be the set of endpoints in $V \setminus S_r$ that show up in edge/round pairs in Z. For each $v \in Y$ we define $Z_v = \{p \in Z \mid p.second = v\}$. Notice that $\hat{Z} = \{Z_v \mid v \in Y\}$ describes a partitioning of Z.

A key property of these equivalence classes is that if $p \in Z_v$ and $p' \in Z_{v'}$, for $v \neq v'$, then p and p' are *not* redundant, as by definition the edge/round pairs have distinct second endpoints ($p.second \neq p'.second$). At this point, we know little about the size or number of these partitions—complicating our ability to bound the number that contain at least one good edge/round pair. This brings us to the second step of our organization of Z in which we greedily pack these equivalence classes into better structured sets we call *buckets* as follows:

1) Initialize k buckets $B_1, B_2, ..., B_k$ to be empty. Initialize set $W \leftarrow \hat{Z}$.
2) Remove the largest class Z_v that remains in W. Let i be the smallest index from $\{1, 2, ..., k\}$ such that B_i contain less than $t/2$ edge/round pairs. If no such i exists, we terminate *successfully.* Else, add every edge/round pair in Z_v to B_i and repeat this step.

We must now show that this procedure will always terminate successfully. To do so, we point out several key properties about our partition of Z. First, we know that in each round $r' \in R$, $|M_{r'}| \geq k$ and therefore $|Y| \geq k$. It follows that there are at least k classes in $\hat{Z}$. We also know that each endpoint in Y can show up at most once per round in that round's matching, so each class can have at most t edge/round pairs in it. Finally, we know each round adds at least k new edge/round pairs to Z, so we have at least $t \cdot k$ such pairs total.

A standard greedy packing argument now establishes successful termination. The key observation is that because we add equivalence classes to buckets in order of largest to smallest, if a given bucket B_i has less than $t/2$ edge/round pairs in it, then all of the classes remaining in set W are of size at most $t/2$. Similarly, no class in $\hat{Z}$ has more than t pairs. It follows that a bucket never has more than t pairs in it. We also know there are at least $t \cdot k$ edges to distribute, so all buckets will receive enough edge/round pairs to exceed the $t/2$ threshold by this procedure before we run out of classes to distribute to buckets.

We are finally ready to analyze the probability of sufficient goodness. Recall, at the beginning of this argument, we fixed an execution prefix through $r-1$ rounds and identified a set S_r of nodes that know $\hat{I}$ be the start of round r. We then divided into buckets a collection of edge/round pairs that describe edges that will occur in the dynamic graph over the next t rounds. Each pair describes a future opportunity for a node in S_r to connect to a node not in S_r. We now analyze the probability that in the t rounds that follow we have at least one good edge/round pair in each bucket—ensuring at least k new nodes learn $\hat{I}$, as required by our lemma. To do so, or a given bucket B_i, let Y_i be the number of good edge/round pairs in B_i. Note that:

$$\begin{aligned} E(Y_i) &= \sum_{p \in B_i} X_{p.edge}(p.round) \\ &\geq (t/2)(1/(4\Delta^2)) \\ &= (c \log n)/8 \end{aligned}$$

As argued, the X indicator variables are independent. It follows that we can apply a Chernoff bound to $E(Y_i)$ to prove that for a sufficiently large constant c (defined with respect to the form of the Chernoff bound we use and the level of high probability needed by the analysis), $Y_i \geq 1$ with high probability. A union bound over the k buckets establishes that with high probability *every* bucket has at least one good edge/round pair in it. Because we did not split any equivalence classes between buckets, it follows that at least k nodes not in S_r connect with nodes in S_r—as required to satisfy the lemma. ■

The following lemma follow from a symmetric version of the proof applied to Lemma VI.4:

Lemma VI.5. *Fix some round $r \geq 1$ such that $|S_r| > n/2$. Let $U_r = V \setminus S_r$. There exists a constant $c \geq 1$ such that with high probability: $|U_{r'}| \leq (1 - \frac{\alpha}{4})|U_r|$, where $r' = r + c \cdot \Delta^2 \cdot \log n$.*

We are now ready to prove Theorem VI.1. The proof below levarages Lemmas VI.4 and VI.5 to make a standard epidemic expansion argument.

Proof (of Theorem VI.1).: Divide rounds into phases of length $c \cdot \Delta^2 \cdot \log n$, where $c \geq 1$ is the constant specified by Lemmas VI.4 and VI.5. For each $I \geq 1$, let r_i be the first round of phase i.

At the beginning of a given phase i, if $S_{r_i} \leq n/2$ then we can apply Lemma VI.4, which tells us that with high probability $S_{r_{i+1}}$ is a factor of $(1 + \alpha/4)$ larger than S_{r_i}. If this occurs, we call the phrase *productive*.

Leveraging Fact II.1, it follows that: $(1 + \frac{\alpha}{4})^t \geq 2^{\frac{t \cdot \alpha}{4}}$. Therefore, after at most $t = (4/\alpha) \log (n/2)$ productive phases, more than $n/2$ nodes know $\hat{I}$. By a union bound, with high probability, the first t phases are all productive. Therefore, with high probability, $S_{r_t} > n/2$.

We can now apply Lemma VI.5, which tells us that with high probability, this set of uninformed nodes (i.e., nodes that do *not* know $\hat{I}$) will shrink by a factor less than or equal to $(1 - \alpha/4)$. Once again we call such a phase productive. Leveraging Fact II.1, it follows: $(1 - \frac{\alpha}{4})^{t'} < e^{-\frac{t' \cdot \alpha}{4}}$. Therefore, after at most $t' = (4/\alpha) \ln (n/2)$ productive phases, all remaining nodes learn $\hat{I}$. By a union bound, with high probability, these next t' phases are all productive. Therefore, with high probability, $S_{r_{t+t'}} = n$.

We have shown that with high probability, after $t + t' = O(\log n/\alpha)$ phases all nodes know $\hat{I}$—at which point, every node has $\hat{I}$ as their leader and will never again change this leader. To conclude the proof, we note that $O(\log n/\alpha)$ phases of $O(\Delta^2 \log n)$ rounds each combine to $O((1/\alpha)\Delta^2 \log^2 n)$ total rounds to solve leader election—as claimed by the theorem statement. ■

A New Bound for PUSH-PULL Rumor Spreading: Notice that our blind gossip leader election algorithm can be directly applied to solve the rumor spreading problem (see Section V) in the mobile telephone model with $b = 0$. In particular, in this setting, it describes the classical PUSH-PULL strategy. In [1], we identified the performance of PUSH-PULL in the mobile telephone model with $b = 0$ as an open question. In this previous work, we proved a lower bound that established its performance would not be efficient, but stopped short of providing a upper bound (due, in part, to the complexity of the dependency issues tamed in our above analysis with the careful deployment of bucketed collections of good edges). Our above analysis, therefore, yields the following corollary which resolves this question:

Corollary VI.6. *PUSH-PULL rumor spreading succeeds with high probability in $O((1/\alpha)\Delta^2 \log^2 n)$ rounds when executed in the mobile telephone model with maximum degree Δ, vertex expansion α, stability factor $\tau \geq 1$ and tag length $b = 0$.*

Analysis Optimality: A time complexity in $\Omega(\Delta^2/\alpha)$ might seem pessimistic as Δ can be as large as n. As a point of comparison, in both the classical telephone model and the mobile telephone model with $b = 1$, rumor spreading requires only $O((1/\alpha)\text{polylog}(n))$ rounds for stable graphs. It is straightforward to show, however, that there exists a stable network (with non-constant Δ and α), in which the blind gossip algorithm requires $\Omega(\Delta^2/\sqrt{\alpha})$ rounds. This does not exactly match our upper bound analysis, but it is close, and more importantly it establishes that the performance of this algorithm is fundamentally slower than the $O((1/\alpha)\text{polylog}(n))$ round complexities of the classical and $b = 1$ models. Notice, this result does not rule out the possibility of faster leader election algorithms in this setting with $b = 0$. It concerns only the performance of the blind gossip strategy.

In more detail, consider a network constructed as follows: arrange $\sqrt{n}$ nodes in a line. Call these nodes $u_1, u_2, ..., u_{\sqrt{n}}$. Connect each u_i to its own collection of $\sqrt{n}$ nodes—resulting in a line of $\sqrt{n}$ stars each consisting of $\sqrt{n}$ points. Fix $I_{u_1} = \hat{I}$ (i.e., the center of the first star in the line has the network's smallest UID). For blind gossip to converge, $\hat{I}$ must propagate down the full line. For $\hat{I}$ to propagate from some u_i to its neighbor $\hat{u}_{i+1}$, however, requires u_i to select u_{i+1} to send its proposal from among its neighbors. This occurs with probability $\approx 1/\Delta$. Furthermore, u_{i+1} must select u_i's proposal. Note that if any point in the star centered on u_{i+1} decides to send a proposal it will send it to u_{i+1} (as it is their only neighbor). Each node decides to send a proposal with probability $1/2$. We therefore expect u_{i+1} to receive $\Theta(\Delta)$ proposals. It selects u_i's proposal also with probability $\approx 1/\Delta$. Progress from u_i to u_{i+1} therefore occurs with probability Δ^{-2}. We expect therefore $\Delta^2\sqrt{n}$ rounds for $\hat{I}$ to propagate down the entire line. Notice, however, that $\alpha > 1/n$ in this network, so this above propagation time is also in $\Omega(\Delta^2/\sqrt{\alpha})$.

VII. Leader Election with $b = 1$ and $\tau \geq 1$

We now consider leader election with $b = 1$. We describe and analyze a new algorithm that leverages this single bit advertisement to achieve potentially large efficiency gains over the blind gossip algorithm of Section VI. The algorithm works for any $\tau \geq 1$ and requires no advance knowledge of τ. It does assume, however, that all nodes start during the same round—allowing them to rely on a global round counter to align groups of rounds in useful ways. In Section VIII, we describe how to modify the below algorithm so that it still works even in a setting where nodes can activate in different rounds and have only local round counters. The algorithm description below references the PPUSH information dissemination strategy. See Section V for a reminder of how this strategy works.

The Bit Convergence Leader Election Algorithm: For each node u, let I_u be u's UID. At the beginning of the execution, each u chooses an *ID tag*, indicated t_u, uniformly from the space 1 to n^β, for some constant $\beta \geq 1$ (fixed in the below analysis). Let $k = \lceil \beta \log n \rceil$ be the number of bits required to describe each ID tag. We call the combination (I_u, t_u) an *ID pair*.

Nodes partition rounds into *groups* of length $2 \log \Delta$. They then partition groups into *phases* consisting of k groups each. In the following, we label the phases $1, 2, ...$, and label the groups in each phase $1, 2, ..., k$. At the beginning of each phase, each node u sets a local pair $(\hat{I}_u, \hat{t}_u) \leftarrow (I', t')$, where (I', t') is the ID pair with the smallest tag t' of all ID pairs it has encountered up to this point. We refer to $\hat{t}_u$ as u's *smallest ID tag* and $(\hat{I}_u, \hat{t}_u)$ as u's *smallest ID pair*. Notice, at the beginning of the first phase, $(\hat{I}_u, \hat{t}_u) = (I_u, t_u)$, by default. If a node u has received more than one ID pair with the same smallest tag, it can break ties with the ordering on the UID element of the pairs. After setting its smallest ID

pair $(\hat{I}_u, \hat{t}_u)$ at the beginning of a phase, node u then sets $leader \leftarrow \hat{I}_u$.

The nodes can now proceed with the k groups that make up the current phase. For each group i of the phase, each node u executes PPUSH as follows: it uses bit position i of the binary encoding of $\hat{t}_u$ as the bit it advertises during PPUSH; if a given u connects with a node v, then they send each other $(\hat{I}_u, \hat{t}_u)$ and $(\hat{I}_v, \hat{t}_v)$, respectively, during their connection. We emphasize that nodes only update their smallest ID pairs at the beginning of each phase. ID pairs received during a phase are stored locally until the next such update.

Analysis Preliminaries: We now introduce several useful pieces of notation. At the start of phase i, let b_i be the most significant bit position such that there exists two nodes u and v where $\hat{t}_u$ and $\hat{t}_v$ differ in position b_i. (For example, $b_i = 2$ indicates that at the start of phase i, all nodes have the same value in the most significant bit of their smallest ID tags, but there are at least two nodes with different values in the second most significant bit.) If all nodes have the same smallest ID tag in phase i, we define $b_i = \perp$. In the following, we call bit b_i the *maximum difference bit* for phase i.

For a given phase i, let S_i be the set of nodes with 0 in bit position b_i of their smallest ID tags, and $U_i = V \setminus S_i$ be the set of nodes with a 1 in this position. Notice, for $b_i \neq \perp$, both S_i and U_i are well-defined and non-empty. Let $f(r) = \Delta^{1/r} \cdot c \cdot r \cdot \log n$ be the approximation factor function fixed in Theorem V.2 in Section V. And finally, let $\hat{\tau} = \min\{\tau, \log \Delta\}$ be the relevant stability for this analysis (performance is not improved as we grow τ past $\log \Delta$).

Before continuing to the main analysis we first prove some important properties about b_i and S_i. At a high-level, the below lemma formalizes the intuition that the maximum difference bit can only grow between phases (as once all nodes have the same bits through a given position in their tags, this cannot change going forward), and that during phases with the same maximum difference bit the set of nodes with 0 in that position can only grow (as a node will never swap its smallest ID tag for a larger tag).

Lemma VII.1. *Fix two phases i and j such that $i \leq j$. The following three properties follow: (1) if $b_i = \perp$ then $b_j = \perp$; (2) if $b_i \neq \perp$ and $b_j \neq \perp$ then $b_i \leq b_j$; and (3) if $b_i = b_j \neq \perp$ then $|S_i| \leq |S_j|$.*

Proof: Let Q_i be the set of smallest ID tags in the network during phase i. It follows from the definition of the algorithm that for $j \geq i$: $Q_j \subseteq Q_i$, and that for every i, $|Q_i| > 1$. We can leverage these straightforward observations to prove the three lemma properties.

To prove the first property, fix some i such that $b_i = \perp$. By definition of $\perp$, it follows that $|Q_i| = 1$. By our above observation on the decreasing and non-empty nature of Q, it follows that $Q_j = Q_i$ and therefore $b_j = \perp$ as well.

To prove the second property, fix some i and j such that $b_i \neq \perp$ and $b_j \neq \perp$. By definition, all tags in Q_i have the same bits in the first $b_i - 1$ bit positions (ordering the bits from most to least significant). By our above observation, $Q_j \subseteq Q_i$ so the same must be true of the tags in Q_j. It follows that $b_j \geq b_i$.

To prove the third property, fix some i and j such that $b_i = b_j \neq \perp$. By definition, every smallest ID tag owned by a node in S_i during phase i is *smaller* than every smallest ID tag owned by a node in U_i during phase i. This follows from the definition of S_i and U_i which tells us that all nodes have the same bits through position $b_i - 1$, and then in position b_i nodes in S_i have a 0 whereas nodes in U_i have a 1. The third property can be restated to say that no node in S_i will later adopt a smallest ID tag currently owned by a node in U_i. Given that a node only adopts a new smallest ID tag if it is smaller than its current tag, and our above observation that all S_i tags are smaller than all U_i tags, it follows that this will never happen. ■

Analysis: Our goal is to prove the following theorem regarding the performance of the bit convergence algorithm in the mobile telephone model:

Theorem VII.2. *The bit convergence leader election algorithm solves the leader election problem in $O\big((1/\alpha)\Delta^{1/\tau}\tau \log^5 n\big)$ rounds when executed in the mobile telephone model with maximum degree Δ, vertex expansion α, stability factor at least τ, $1 \leq \tau \leq \log \Delta$, and tag length $b = 1$.*

We begin by studying the spread of small ID tags in the network. To do so, fix some phase i such that $b_i \neq \perp$. By definition, the bit convergence leader election algorithm executes PPUSH during group b_i of this phase with the nodes in S_i acting as the informed nodes and those in U_i acting as the uninformed nodes. Similar to our analysis of rumor spreading in [1], we call this phase *good* if we grow S_i (or, equivalently, shrink U_i) by a sufficient magnitude, where in this context we define "sufficient" with respect to the graph's vertex expansion α and the approximation factor $f(\hat{\tau})$ defined above in the analysis preliminaries.

Definition VII.3. *Fix some phase i with $b_i \neq \perp$. We consider two cases for considering a phase* good*:*

- *If $|S_i| \leq n/2$, we call this phase* good *if: (1) $b_{i+1} \neq b_i$; or (2) $|S_{i+1}| \geq \big(1 + \frac{\alpha}{4 \cdot f(\hat{\tau})}\big)|S_i|$.*
- *Else if $|S_t| > n/2$, we call this phase* good *if (1) $b_{i+1} \neq b_i$; or (2) $|U_{i+1}| \leq \big(1 - \frac{\alpha}{4 \cdot f(\hat{\tau})}\big)|U_i|$.*

In our analysis of the bit convergence algorithm, the core unit of progress is advancing maximum bit difference values. This advancement matters because these values can only increase a bounded number of times before it must be the case that all nodes have converged to the same smallest ID tag (which, under the assumption that these

tags are unique, implies convergence to a single leader). The following lemma bounds the number of good phases required to guarantee the maximum bit difference increases. Notice, the below proof leverages Lemma VII.1 to ensure we do not backtrack between good phases. It also uses the definition of $\hat{\tau}$ from the analysis preliminaries.

Lemma VII.4. *Fix some phase i such that $b_i \neq \perp$. Let $t_{max} = \lceil (1/\alpha) 8 f(\hat{\tau}) \log n \rceil$. Assume there are at least t_{max} good phases between phase i and some phase $j \geq i + t_{max}$. It follows that either $b_j = \perp$ or $b_j > b_i$.*

Proof: Fix some phase i as specified by the lemma statement. By Lemma VII.1, once we arrive at any phase j such that $b_j = \perp$ or $b_j > b_i$, we are done, as this will necessarily be true of all future phases. To count how many good phases are required to guarantee this occurs, we notice that for each phase $i' \geq i$ such that $b_{i'} = b_i$, there are two cases for what happens if i' is good. These case depend on the size of $S_{i'}$:

- The *first case* applies if the number of nodes with a 0 in position $b_{i'}$ is no more than $n/2$ (i.e., $|S_{i'}| \leq n/2$). By Definition VII.3, either $b_{i'+1} = \perp$, or the number of nodes with a 0 in this position grows by a factor greater than or equal to $\left(1 + \frac{\alpha}{4 \cdot f(\hat{\tau})}\right)$.
- The *second case* applies if the number of nodes with a 0 in position $b_{i'}$ is greater than $n/2$ (i.e., $|S_{i'}| > n/2$). By Definition VII.3, either $b_{i'+1} = \perp$, or the number of nodes with a 1 in this position shrinks by a factor less than or equal to $\left(1 - \frac{\alpha}{4 \cdot f(\hat{\tau})}\right)$.

Leveraging Fact II.1, it follows that:

$$\left(1 + \frac{\alpha}{4 \cdot f(\hat{\tau})}\right)^{t_{grow}} \geq 2^{\frac{t_{grow} \cdot \alpha}{4 \cdot f(\hat{\tau})}}.$$

The first case from above, therefore, can occur at most $t_{grow} = (1/\alpha) 4 f(\hat{\tau}) \log (n/2)$ times before either we arrive at a phase i' with $i' = \perp$, or we have grown the size of $S_{i'}$ to a size of at least $n/2$ and we are no longer in the first case. Crucial to this argument is the third property of Lemma VII.1, which tells us that the $S_{i'}$ set cannot shrink. Therefore, in between good phases, we do not lose ground, and once move to the second case we cannot move back to the first.

After this point, we turn our attention to the second case. Leveraging Fact II.1, it follows:

$$\left(1 - \frac{\alpha}{4 \cdot f(\hat{\tau})}\right)^{t_{shrink}} < e^{-\frac{t_{shrink} \cdot \alpha}{4 \cdot f(\hat{\tau})}}.$$

The second case from above, therefore, can occur at most $t_{shrink} = (1/\alpha) 4 f(\hat{\tau}) \ln (n/2)$ times before we arrive at a phase i' where either $b_{i'} = \perp$, or we have reduced the number of nodes with a 1 in position b_i to 0. This later event implies $b_{i'} > b_i$. As above, we leverage the third property of Lemma VII.1 to ensure that $U_{i'}$ cannot grow between good phases. To conclude the lemma, we note $t_{max} > t_{grow} + t_{shrink}$. ∎

The properties studied so far have been deterministic. We now turn to the probabilistic nature of the algorithm by lower bounding the probability that a given phase is good. This argument leverages Theorem V.2 from Section V which describes the effectiveness of PPUSH for a bounded number of stable rounds.

Lemma VII.5. *There exists a constant probability $p_g > 0$ such that for any phase i with $b_i \neq \perp$, the probability that phase i is good is at least p_g.*

Proof: Fix some phase i as specified by the lemma statement. Consider group b_i in phase i. Recall that $\hat{\tau} = \min\{\tau, \log \Delta\}$. Because each group consists of $2 \log \Delta$ rounds, it follows that there must be a stretch of $\hat{\tau}$ consecutive *stable* rounds in this group (i.e., rounds in which the graph does not change). Let G_i be stable graph during these $\hat{\tau}$ consecutive rounds in group b_i of phase i.

Now we study the properties for G_i. In particular, let M_i be a maximum matching between S_i and U_i in G_i. Formally, M_i is a maximum matching in $B(S_i)$ defined with respect to G_i (see Section V for the formal definition of B). Let $m = |M_i|$ be the size of this matching.

We consider two cases with respect to the size of S_i. The first case is that $|S_i| \leq n/2$. In this case, by Lemma V.1 in Section V applied to G_i, it follows that $m/|S_i| \geq \alpha/4 \Rightarrow m \geq |S_i| \cdot (\alpha/4)$.

We now consider how many pairs in this matching of size m we expect to successfully connect in the $\hat{\tau}$ rounds during which the graph remains stable as G_i. To then end, we deploy Theorem V.2 from Section V. In more detail, we apply this theorem where $L \subseteq S_i$ contains all nodes in S_i that are endpoints of an edge in the matching M_i, R contains the neighbors of L in G_i that are also in U_i, G is the bipartite graph with bipartitions L and R, and an edge set $\{\{u, v\} \mid u \in L, v \in R, \{u, v\} \in G_i\}$, and $r = \hat{\tau}$. It follows from Theorem V.2 applied to these parameters that there is a constant probability p, such that with probability at least p, at least $|S_i| \cdot (\alpha/4) \cdot (1/f(\hat{\tau}))$ nodes in U_i connect with a node from S_i (and therefore shift to S_{i+1}). Put another way, with probability at last p, $|S_i|$ grows by a factor of at least $\left(1 + \frac{\alpha}{4 \cdot f(\hat{\tau})}\right)$ between phase i and $i+1$—exactly matching the first case of our definition of *good* (Definition VII.3).

The second case to consider is when $|S_i| > n/2$. Here we can apply the same argument as for the first case, with the exception that now $m \geq |U_i| \cdot (\alpha/4)$. The result is that with in this case, with probability at least p, $|U_i|$ shrinks by a factor of $\left(1 - \frac{\alpha}{4 \cdot f(\hat{\tau})}\right)$ between phase i and $i+1$—exactly matching the second case of our definition *good* (Definition VII.3). Combining these two cases it is clear that the lemma holds for probability $p_g = p$. ∎

We are now ready to prove our main theorem. The core insight is that there are only k bit positions in an ID tag.

Therefore, we can only increase the maximum bit difference k times before all nodes have the same ID tag and the leader election problem is solved (assuming that all initial choices of ID tags are unique). We can combine Lemmas VII.4 and VII.5, with a stochastic dominance argument (to handle dependencies between phases) to bound the number of rounds needed to achieve the needed number of advances with a sufficiently high probability.

Proof (of Theorem VII.2): We begin by assuming that at the beginning of the execution each node selects a unique ID tag. This occurs with high probability in n that grows with the multiplicative constant β in the definition of k.

We now calculate how many phases are needed to ensure at least t_{max} (from Lemma VII.4) are good, with high probability. To do so, for any given phase t, let X_t be the random indicator variable that evaluates to 1 if phase t is good (or $b_t = \perp$), and otherwise evaluates to 0. For any given integer $T > 0$, and phase $i > 0$, let $Y_{T,i} = \sum_{t=i}^{i+T-1} X_t$ be the number of good (or already converged) phases in the T phases $i, i+1, ..., i+T-1$. We know from Lemma VII.5 and linearity of expectation that $E(Y_{T,i}) \geq p_g T$. We cannot directly concentrate on this expectation, however because X_t and $X_{t'}$ might be dependent for $t \neq t'$.

To overcome this issue, for each phase t, fix $\hat{X}_t$ to be the trivial random variable that evaluates to 1 with independent probability p_g, and otherwise evaluates to 0. By Lemma VII.5 we know that $Pr(X_t = 1) \geq p_g$, regardless of the behavior in previous phases It follows that for every t, X_t stochastically dominates $\hat{X}_t$. Accordingly, if $\hat{Y}_{T,i} = \sum_{t=i}^{i+T-1} \hat{X}_t$ is greater than some x with some probability $\hat{p}$, then $Y_{T,i}$ is greater than x with probability at least $\hat{p}$.

A Chernoff bound applied to $\hat{Y}_{T,i}$, for any phase i and $T = c \cdot t_{max}$ (where $c \geq 1$ is a sufficiently large constant defined with respect to constant p_g and the Chernoff form deployed), provides that $\hat{Y}_{T,i} \geq t_{max}$ with high probability in n. It follows the same holds for $Y_{T,i}$.

We have established, therefore, that with high probability, every $\Theta(t_{max})$ phases we experience at least t_{max} good phase. By Lemma VII.4, this is a sufficient number of good phases to ensure that the maximum difference bit either increases or converges to $\perp$. We can advance the maximum difference bit at most $k = \Theta(\log n)$ times before it converges to $\perp$. Therefore, applying a union bound to the (at most) k advances, and the assumption that all ID tags are unique, it follows that with high probability (with an exponent that grows with constants β and c) our algorithm converges to a single unique ID in at most:

$$\begin{aligned} O\big(t_{max} \log n\big) &= O\big((1/\alpha) f(\hat{\tau}) \log^2 n\big) \\ &= O\big((1/\alpha) \Delta^{1/\hat{\tau}} \hat{\tau} \log^3 n\big) \end{aligned}$$

phases. To obtain our final time complexity result, we note that each phase consists of $2k \log \Delta \in O(\log^2 n)$ rounds. ■

VIII. Leader Election with Asynchronous Activations

The bit convergence leader election algorithm described and analyzed in Section VII assumes all nodes start during the same round. The assumption simplifies matters as it provides nodes synchronized round counters, which allow them to assign rounds to specific bits from their ID tags in a consistent manner.

Here we consider the harder case where nodes might activate during different rounds. In more detail, we assume that each node begins with a local round counter initialized to 1 when it activates, and is provided no *a priori* knowledge of the other nodes' activation statuses or local round counters. In this setting, we modify the definition of leader election so that when we claim an algorithm solves the problem in T rounds, we mean that it solves the problem in T rounds *after* the last node activates.

Below we describe a modification to our bit convergence algorithm that solves leader election in the asynchronous activation setting with a round complexity that is within polylogarithmic factors of the original algorithm. The algorithm requires an advertising tag length $b = \log \log n + O(1)$. Our original algorithm, by contrast, can work for any $b \geq 1$. It remains an open question whether this small gap can be closed in the asynchronous activation setting without a major impact on the time complexity.[2]

The Non-Synchronized Bit Convergence Leader Election Algorithm: Here we describe our modifications to bit convergence algorithm from Section VII. As in the original algorithm, nodes randomly generate ID tags containing $k = \beta \log N$ bits (for some constant $\beta \geq 1$ fixed in the analysis) to pair with their UIDs, and keep track of the smallest ID pair they have received so far in the execution. Also as in the original algorithm, nodes divide their rounds into *groups* consisting of $2 \log \Delta$ rounds each. Notice, however, unlike the original algorithm, group boundaries are not necessarily synchronized between different nodes as they can now activate at different rounds.

Each node u, at the beginning of each of its groups, selects a bit position $i \in [k]$ with uniform randomness. During all $2 \log \Delta$ rounds of the this group, u advertises the position i, *as well as* the value of the bit in position i of the ID tag of its current smallest ID pair. Notice, advertising i requires up to $\log k$ bits (as there are k bit positions). One extra bit is required to describe the bit in position i. Therefore, any tag length $b > \lceil \log k \rceil = \log \log n + O(1)$ is sufficient.

[2]We already know it is *feasible* to solve leader election in the asynchronous activation setting with small b values—if time complexity is not important. In more detail, our blind gossip leader election algorithm assumes $b = 0$ and makes no assumption about round synchronization. Its analysis, therefore, directly applies to the asynchronous activation setting. Its time complexity, however, includes an extra Δ factor. The true open question is whether it is possible to solve leader election in this setting in similar time to blind convergence with $b = O(1)$.

Fix some group during which node u is advertising the bit in position i. During this group, u runs a slightly modified version of the PPUSH information spreading strategy used in the original algorithm. In particular, if u is advertising a 1 bit in position i, it receives connection proposals during the rounds of the group. On the other hand, if u is advertising a 0 bit for position i, it sends PPUSH connection proposals during the rounds of this group. In more detail, in each round, it chooses a recipient for a connection proposal uniformly from neighbors that: (1) are also advertising position i; and (2) advertise a 1 in that bit position (if any such neighbors happen to exist). In other words, nodes only want to deal with other nodes that happen to be advertising the same ID tag bit position in that round.

If two nodes u and v connect, they behave the same as in the original algorithm: they trade smallest ID pairs, and update their locally stored smallest ID pair if the pair they received is smaller than what they are currently storing.

Analysis: The analysis that follows modifies the existing analysis of the original bit convergence algorithm from Section VII. Accordingly, we reference much of the existing terminology, and several proof arguments, from this section without recreating them from scratch here.

We also, however, introduce some new notation useful for studying our new modifications. For each node u and each round $r \geq 1$, let $\hat{t}_u^{(r)}$ be the tag in u's smallest ID pair at the start of round r. Let $\hat{t} = \min_{u \in V}\{\hat{t}_u^{(1)}\}$ be the smallest ID tag in the system at the start of the execution. For a given tag t and position $i \in [k]$, let $t[i]$ be the bit in position i of tag t. Recall, that we use the convention that in interpreting tags as numerical values, we list the bits in decreasing order of significance: that is, for tag t, $t[1]$ is the most significant bit and $t[k]$ is the least.

We begin our analysis by establishing a useful property of our algorithm: if the $i \in [k]$ most significant bits of the tag in a node's smallest ID pair are the same as these corresponding bits in $\hat{t}$, this node, going forward, will never change this tag to one with different bits in these positions. This property, which we formalize below, follows from the definition of our algorithm which only allows a node to replace its smallest ID pair if it encounters one with a smaller ID tag (or the same tag and smaller UID). If the i most significant bits of a node's tag are the same as these bits in $\hat{t}$, then any tag with a different set of bits in these positions must be larger.

Lemma VIII.1. *Fix some node u, round $r \geq 1$, and tag bit position $i \in [k]$. Assume $\forall j \in [i] : t_u^{(r)}[j] = \hat{t}[j]$. Then for every $r' \geq r : \forall j \in [i] : t_u^{(r')}[j] = \hat{t}[j]$.*

We are now ready for our main theorem. The proof that follows will study the spread of bits from $\hat{t}$, one at a time, starting with the most significant bit, and leveraging Lemma VIII.1 to consolidate gains. We will modify the analysis of the original bit convergence algorithm to bound the time required for these spreads given the non-synchronized conditions of this setting. The extra round complexity cost, roughly speaking, covers the new need for relevant neighbors to both randomly select the right bit positions to advertise.

Theorem VIII.2. *The non-synchronized bit convergence leader election algorithm solves the leader election problem in $O\big((1/\alpha)\Delta^{1/\tau}\tau\log^8 n\big)$ rounds after the last node is activated when executed in the mobile telephone model with asynchronous activations, maximum degree Δ, vertex expansion α, stability factor at least τ, $1 \leq \tau \leq \log\Delta$, and tag length $b = \lceil \log k \rceil + 1 = \log\log n + O(1)$.*

Proof: Assume that at the start of the execution every node generates a unique ID tag. This occurs with high probability that we can adjust with the constant in the definition of k. We begin our analysis by considering the time required for every node to adopt a tag with bit $\hat{t}[1]$ in the first (most significant) position of the tag in its smallest ID pair (which we also call its "smallest ID tag" in the following), after all nodes are activated. By Lemma VIII.1, once any node adopts a smallest ID tag with the same bit as $\hat{t}$ in its first position, it will never again adopt a tag with a different bit in that position. We can, therefore, treat the spread of tags with this bit like a spreading rumor, and bound the time for this spread to complete using the analysis from our original bit convergence algorithm.

To apply this existing analysis, however, will require some new scaffolding to compensate for the lack of synchronization between nodes. With this in mind, fix some arbitrary node u, and some integer constant $c > 1$ that we will bound later in this analysis. Consider u's local group schedule. We use the notation $g(i)$, for integer $i \geq 1$, to indicate the i^{th} full group in u's schedule that occurs *after* all nodes are activated. We now select a subset of these groups to play the role of *reference groups* in our analysis. In more detail, $g(c)$ is reference group 1, $g(2c)$ is reference group 2, and, more generally, for every $i \geq 1$: $g(i \cdot c)$ is reference group i. That is, we select one out of every c of u's groups to play the role of a reference group in our analysis.

Let S_i, at the beginning of a given round, describe the nodes with $\hat{t}[i]$ in position i of their tag. In this stage of the analysis, we are bounding the number of reference groups needed until S_1 includes all the nodes. To do so, we will adapt the definition of *good* from the original analysis to apply to our reference groups (instead of phases). We will also modify its definition to reduce the fraction by an extra k^4 term. That is, we call a reference group good if the relevant set grows or shrinks by a factor of $\frac{\alpha}{c' \cdot f(\hat{\tau}) \cdot k^4}$, for some constant $c' \geq 1$ we fix below.

Let t_{max} be the number of good phases needed in the worst case before S_1 includes all nodes. It is straightforward to adjust our analysis of t_{max} from Lemma VII.4 to compensate for this extra factor, as this analysis depends only on the fraction used in the definition of good and does

not depend on the details of the underlying algorithm. The new result now states that $t_{max} = O((1/\alpha)f(\hat{\tau})k^4 \log n)$

We must now adapt Lemma VII.5 to establish that the probability Following the same general outline as this existing proof (but now shifting our attention from phases to reference groups), fix some reference group i. Because this group contains $2 \log \Delta$ rounds, there must be some stretch of $\hat{\tau} = \min\{\tau, \log \Delta\}$ consecutive rounds in this group during which the network topology does not change. Let G_i be the stable graph during these $\hat{\tau}$ consecutive rounds.

As in the original analysis, we can identify a large matching M_i in G_i between nodes in S_1 and nodes not in S_1. Let $m = |M_i|$. In the original analysis, we could assume that all endpoints in M_i would be running PPUSH with the same bit position during the stable rounds. This assumption does not hold for our modified algorithm.

To handle this reality, we say a given node $v \in M_i$ is *useful* if it is advertising bit position 1 for every round of this reference group. Notice, even though v's local groups are not necessarily synchronized to the reference group's boundaries, it takes at most two consecutive v groups to cover all the rounds of the rounds of our fixed reference group. For v to be useful, therefore, requires (at most) that it selects bit position 1 for two of its groups in a row. It follows that node v is useful with probability at least $1/k^2$. Therefore, the probability that both endpoints of a given edge in M_i are useful is at least $1/k^4$.

Let $\hat{M}_i$ be the subset of M_i that is useful, and $\hat{m} = |\hat{M}_i|$. We can continue the Lemma VII.5 analysis with $\hat{M}_i$, which tells us that with at least some constant probability p_1, at least $\hat{m}/f(\hat{\tau})$ nodes in this matching connect and learn a tag with $\hat{t}[1]$ in the first position.

Because the choice of bit to advertise in a group is made with independent randomness, it is straightforward to show that with some constant probability p_2, $\hat{m}$ is within a constant fraction of its expected value of m/k^4. That is, it is at least $c'' \cdot (m/k^4)$ for some constant $c'' > 0$. It follows that with constant probability $p_1 p_2$: at least $(c''m)/(k^4 f(\hat{\tau}))$ new nodes learn the bit.

Given the bounds on the size of M_i derived from the original analysis, this is a sufficient number of new nodes entering S_1 for the reference group to satisfy our modified definition of good (assuming we adjust the definition of c' used in the good definition to appropriately accommodate constant c'').

We can now focus on how many reference groups are needed before t_{max} are good with high probability. In this case, we can apply the same stochastic dominance argument used in the proof of the main theorem in the analysis of the original algorithm. The only new care that must be taken is to ensure that node u's relevant bit choices for a given reference group do not intersect the relevant choices for a different group. This is easily accomplished if we set $c > 2$ in our selection of the reference groups. This ensures that each pair of reference groups are separated by at least two non-reference groups. This in turn ensures that the groups a node must use to cover a given reference group are different for each reference group.

To conclude this proof, we note that we have established that with high probability, all nodes have $\hat{t}[1]$ in position 1 of their tag after $O(t_{max})$ reference groups. By Lemma VIII.1, this bit will never again change so we can now apply the same analysis to $\hat{t}[2]$, then $\hat{t}[3]$, and so on, until all nodes have exactly $\hat{t}$ as their smallest ID tag. Combined with our original assumption that all ID tags are unique, it follows that we solved leader election. A union bound can be used to combine the k instances of the above argument as well as the unique ID tag assumption. Since all hold with high probability the result remains high probability.

Putting together the pieces, we have shown that with high probability, $O(kt_{max})$ reference groups are sufficient to solve the problem. Each group requires $\Theta(\log \Delta)$ rounds (including the constant number of non-reference groups that separate each reference group). Our total time complexity, therefore, can be bounded as

$$O(k \log \Delta t_{max}) = O((1/\alpha)f(\hat{\tau})k^5 \log n \log \Delta) =$$

$$O((1/\alpha)\Delta^{1/\hat{\tau}} \hat{\tau} \log^8 n),$$

rounds, as required by the theorem statement. ■

IX. Conclusion

In this paper, we study leader election in the mobile telephone model. We prove that efficient leader election is possible in this model given small advertising tags and reasonable stability, where we define "efficient" to mean within polylogarithmic factors of the classical telephone model results. This work also generated several interesting open problems. Most of our algorithms, for example, include a $\Delta^{1/\tau}$ term that decreases from Δ to 1 as the stability factor τ increases from 1 to $\log \Delta$. It is unclear whether this *cost of mobility* is fundamental or if more clever algorithmic strategies might achieve better performance for small τ. Another open question is the relationship between the tag length b and leader election performance. In this paper, shifting from $b = 0$ to $b = 1$ allowed us to solve leader election significantly more efficiently, and the shift from $b = 1$ to $b = \log \log n + O(1)$ allowed us to also handle asynchronous activations. We have no lower bounds establishing any of these gaps as fundamental. Investigating the power of advertisements remains a key question about the mobile telephone model. Beyond these open questions, we emphasize that the model itself is well-motivated and can be used to study any number of other problems that might prove useful in a peer-to-peer setting, including, for example, gossip, consensus, and data aggregation.

REFERENCES

[1] M. Ghaffari and C. Newport, "How to discreetly spread a rumor in a crowd," in *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2016.

[2] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," *Sensors*, vol. 12, no. 9, pp. 11 734–11 753, 2012.

[3] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with wi-fi direct: overview and experimentation," *IEEE wireless communications*, vol. 20, no. 3, pp. 96–104, 2013.

[4] D. Mark, J. Varma, J. LaMarche, A. Horovitz, and K. Kim, "Peer-to-peer using multipeer connectivity," in *More iPhone Development with Swift*. Springer, 2015, pp. 239–280.

[5] N. Cohen, "Hong Kong Protests Propel FireChat Phone-to-Phone App," *The New York Times*, July 5, 2014, available online: http://www.nytimes.com/2014/10/06/technology/hong-kong-protests-propel-a-phone-to-phone-app-.html .

[6] A. M. Frieze and G. R. Grimmett, "The shortest-path problem for graphs with random arc-lengths," *Discrete Applied Mathematics*, vol. 10, no. 1, pp. 57–77, 1985.

[7] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking, "Randomized rumor spreading," in *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*, 2000, pp. 565–574.

[8] G. Giakkoupis, "Tight bounds for rumor spreading in graphs of a given conductance," in *Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS)*, 2011.

[9] F. Chierichetti, S. Lattanzi, and A. Panconesi, "Rumour spreading and graph conductance." in *Proceedings of the ACM-SIAM symposium on Discrete Algorithms (SODA)*, 2010.

[10] G. Giakkoupis and T. Sauerwald, "Rumor spreading and vertex expansion," in *Proceedings of the ACM-SIAM symposium on Discrete Algorithms (SODA)*. SIAM, 2012, pp. 1623–1641.

[11] N. Fountoulakis and K. Panagiotou, "Rumor spreading on random regular graphs and expanders," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2010, pp. 560–573.

[12] G. Giakkoupis, "Tight bounds for rumor spreading with vertex expansion," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014.

[13] S. Daum, F. Kuhn, and Y. Maus, "Rumor spreading with bounded in-degree," *arXiv preprint arXiv:1506.00828*, 2015.

[14] M. K. Aguilera, C. Delporte-Gallet, H. Fauconnier, and S. Toueg, "Stable leader election," in *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2001.

[15] B. Awerbuch, "Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems," in *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 1987.

[16] S.-T. Huang, "Leader election in uniform rings," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 15, no. 3, pp. 563–573, 1993.

[17] G. Singh, "Leader election in the presence of link failures," *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, no. 3, pp. 231–236, 1996.

[18] M. K. Aguilera, C. Delporte-Gallet, H. Fauconnier, and S. Toueg, "Communication-efficient leader election and consensus with limited link synchrony," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*. ACM, 2004, pp. 328–337.

[19] M. Ghaffari and C. Newport, "Leader election in unreliable radio networks," in *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, 2016.

[20] F. Kuhn, N. Lynch, and R. Oshman, "Distributed computation in dynamic networks," in *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 2010.

[21] N. Malpani, J. L. Welch, and N. Vaidya, "Leader election algorithms for mobile ad hoc networks," in *Proceedings of the International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*. ACM, 2000.