Improved Fault Tolerance and Secure Computation on Sparse Networks

Nishanth Chandran[1]* Juan Garay[2] Rafail Ostrovsky[3]**

[1] Department of Computer Science, UCLA. Email: nishanth@cs.ucla.edu
[2] AT&T Labs – Research, Florham Park, NJ, Email: garay@research.att.com
[3] Departments of Computer Science & Mathematics, UCLA
Email: rafail@cs.ucla.edu

Abstract. In the problem of *almost-everywhere agreement* (denoted a.e. agreement), introduced by Dwork, Peleg, Pippenger, and Upfal [STOC '86], n parties want to reach agreement on an initially held value, despite the possible disruptive and malicious behavior of up to t of them. So far this is reminiscent of the classical *Byzantine agreement* problem, except that in the alternative formulation the underlying connectivity graph is *sparse*—i.e., not all parties share point-to-point reliable channels—thus modeling the actual connectivity of real communication networks. In this setting, one may not be able to guarantee agreement amongst all honest parties, and some of them, say x, must be given up. Thus, in this line of work the goal is to be able to tolerate a high value for t (a constant fraction of n is the best possible) while minimizing x. As shown in the original paper, the dependency on d, the degree of the network, to achieve this goal is paramount.

Indeed, the best polynomial-time a.e. agreement protocol tolerating a constant fraction of corruptions $t = \alpha n$, for some constant $\alpha > 0$ (presented in the original paper, over two decades ago) has parameters, $d = n^{\epsilon}$ for constant $\epsilon > 0$ and $x = \mu t$ for some constant $\mu > 1$. In this work, we significantly improve on the above parameters obtaining a protocol with $t = \alpha n$, $d = \mathcal{O}(\log^q n)$, for constant $q > 0$ and $x = \mathcal{O}(\frac{t}{\log n})$. Our approach follows that of Dwork *et al.* of reducing the a.e. agreement problem to constructing a reliable transmission scheme between pairs of nodes for a large fraction of them; however, given our setting's more stringent conditions—poly-logarithmic degree and a linear number of corruptions, such a task is significantly harder.

We also consider the problem of *secure computation* on partially connected networks, as formulated by Garay and Ostrovsky [Eurocrypt '08], and as a corollary to our main result obtain an almost-everywhere secure multi-party computation protocol with the same parameters as above, again significantly improving on the bound on the number of left-out parties—$x = \mathcal{O}(\frac{t}{\log n})$ for $t \leq \alpha n$ corruptions, as opposed to $x = \mathcal{O}(t)$ in the original work.

* Work partly done while visiting AT&T Labs – Research. Supported in part by NSF grants 0716835, 0716389, 0830803, 0916574

** Supported in part by IBM Faculty Award, Xerox Innovation Group Award, the Okawa Foundation Award, Intel, Teradata, NSF grants 0716835, 0716389, 0830803, 0916574 and U.C. MICRO grant.

1 Introduction

Clearly, partially connected networks are far more realistic than fully connected networks, especially as the number of nodes grows. Indeed, essentially all deployed practical networks do not have dedicated links between each pair of nodes, and instead rely on routing and multi-hop message transmission protocols. In a seminal paper, Dwork, Peleg, Pippenger and Upfal [10] introduced the notion of *almost-everywhere agreement* (a.e. agreement for short) in an effort to model the reality of communication networks, as well as to capture the impact of limited connectivity on the properties of fundamental fault-tolerant distributed tasks, such as *Byzantine agreement* [21, 18], which requires that parties agree on a value based on initial inputs held by each of them, despite the disruptive behavior of some of the nodes, and which so far had only been studied assuming full connectivity amongst the parties.

Instead, in the Dwork *et al.* formulation, the n parties, or nodes, are connected by a graph G. Nodes that are connected by an edge in G share a point-to-point reliable channel with each other; other nodes have to communicate by transmitting messages over the paths that might be available to them, through other nodes in the network. Clearly, in such a setting, one may not be able to guarantee agreement for all nodes in the network, even if they are honest. For example, if an honest node is connected only to misbehaving (or "corrupt") nodes, then guaranteeing that this node reaches agreement with other honest nodes is impossible. In other words, these honest nodes have to be "given up" (hence the term "almost-everywhere" agreement) and its number becomes a new relevant parameter to the problem. Thus, the added goal of fault-tolerant protocols for partially connected settings, besides tolerating the maximal number of faults or corruptions, is to minimize the number of such excluded parties—a task which, as shown in [10], heavily depends on the degree of the graph.

Indeed, Dwork *et al.* constructed a.e. agreement protocols for several classes of partially connected networks, which make the interplay between the various parameters apparent. For example, they presented a protocol for a.e. agreement on a constant-degree graph tolerating up to $t = \mathcal{O}(\frac{n}{\log n})$ corrupt nodes, while guaranteeing that no more than $x = \mathcal{O}(t)$ are left out. They also constructed a protocol on a graph of degree n^{ϵ} for constant ϵ tolerating up to $t = \mathcal{O}(n)$ corrupt nodes, with $x = \mathcal{O}(t)$. The main problem left open in [10] was to construct graphs of low degree tolerating $t = \mathcal{O}(n)$ corrupt nodes with $x = \mathcal{O}(t)$ (or even lower) excluded nodes. In a remarkable result, Upfal [22] showed the existence of an a.e. agreement protocol on constant-degree graphs tolerating a constant fraction of corrupt nodes, while giving up a linear (in t) number of honest nodes. Unfortunately, the protocol of [22] runs in exponential time (in n).

In this work, we construct graphs of poly-logarithmic degree ($\mathcal{O}(\log^q n)$, for constant $q > 0$), on which we show an efficient (i.e., polynomial-time) a.e. agreement protocol tolerating up to a constant fraction of corrupt nodes; further, and in contrast to previous results, we only give up $\mathcal{O}(\frac{t}{\log n})$ nodes, as opposed to linear (in t). Thus, our protocol is the first significant (in fact, *exponential*) improvement on the degree of graphs, as well as on the number of given-up nodes from a constant fraction to a sub-constant, admitting deterministic polynomial-time protocols since the work of [10] more than 20 years ago. We remark that, similarly to [22], we construct a specific graph of poly-logarithmic degree along with an a.e. agreement protocol on this graph network.

We also consider the problem of (unconditional, or information-theoretic) *secure multi-party computation* (MPC) [2, 8] in the context of partially connected networks, as formulated by Garay and Ostrovsky [13]. By applying our new a.e. agreement protocol to the construction of [13], we immediately obtain an a.e. MPC protocol for graphs of poly-logarithmic degree, tolerating a linear number of corruptions, while giving up only $\mathcal{O}(\frac{t}{\log n})$ parties from the secure computation.

Related work. We already mentioned above the formulation of the a.e. agreement problem by Dwork *et al.* [10], and the result, albeit inefficient, by Upfal [22] for constant-degree graphs tolerating a constant fraction of corruptions and yet guaranteeing agreement also for a constant fraction of the nodes. Berman and Garay [4, 5], improved the efficiency of the protocols from [10], while Ben-Or and Ron [3] considered the problem in which faults are random and not adversarially chosen. Bagchi *et al.* [1] considered a related problem of obtaining a large connected component of a graph (that avoids all adversarial nodes), such that this connected component has high expansion given that the original graph had high expansion. This result can be applied to improve the bounds obtained by Upfal [22]. However, the algorithm for obtaining this large connected component also does not run in polynomial time (in addition to the exponential-time protocol from [22] that one would have to run on this connected component).

An alternative view to limited network connectivity is to try to minimize the communication costs of fault-tolerant distributed tasks ($\Omega(n^2)$ in the case of Byzantine agreement [9]) for scalability purposes. In that vein, King *et al.* [17] constructed a *randomized* a.e. agreement protocol with low communication and parameters similar to ours, i.e., their protocol tolerates linear number of corruptions and gives up $\mathcal{O}(\frac{t}{\log n})$ honest nodes. However, besides being non-error-free and requiring parties to have access to private random bits (something unavoidable if the communication complexity lower bound is to be overcome), the protocol is only able to tolerate *static* corruptions (meaning which parties are corrupted must be specified before the protocol execution starts); in contrast, our protocol enjoys *adaptive* security, allowing an adversary to use information obtained from a set of corrupted parties at one round to determine which node(s) to corrupt next. In the fully connected network model, further work by King and Saia [16] builds on [17] in the sense of avoiding all-to-all communication, and yet achieve full agreement at a lower communication cost. The work of [16] also differs from our setting in the following ways – it considers a fully connected network, utilizes private random bits, and considers only static corruptions.

Overview of our techniques. We will call the honest nodes for which we guarantee agreement the *privileged* nodes, and the honest nodes for which we do not the *doomed* nodes. Effectively, we follow the general approach of [10] and [22] for a.e. agreement by constructing a reliable remote message transmission scheme between any two nodes in a large set of privileged nodes. However, given only poly-logarithmic degree and a linear number of corruptions, this is significantly harder than in [10]. On the other hand, the technique used by Upfal [22] is to simply "flood" all the paths with the message, and showing that at least one uncorrupted path exists; one can then exhaustively search for the set of corrupted nodes and obtain the message m, which leads to an exponential-time algorithm. In contrast, we will need to obtain polynomially many good paths between two privileged nodes using just poly-logarithmic degree, even though a linear number of nodes may be corrupted.

Further, unlike the transmission schemes of [10] and [22], nodes along the path in our scheme will be more "proactive" and will not just simply forward the message being sent. This is necessary in order to ensure that the correct message keeps being transmitted along majority of the paths at the different stages. This will then enable us to get a polynomial-time reliable message transmission scheme, which in turn will be sufficient both for our result on a.e. agreement as well as for a.e. MPC.

Our starting point is the reliable remote message transmission scheme $\mathrm{TS}_{\mathrm{dppu}}$ from [10] mentioned above for constant degree networks, which tolerates $t = \mathcal{O}(\frac{n}{\log n})$ number of corruptions and gives reliable communication between any two nodes from a set of privileged nodes of size $n - \mathcal{O}(t)$. At a high level, the scheme associates with every node u in the graph a *fan-in* set and a *fan-out* set of a fixed (but not necessarily constant) size. In addition, (not necessarily vertex-disjoint) paths from a node to its sets are specified, as well as (vertex-disjoint) paths for all ordered pairs of one node's fan-out set to any other node' fan-in set. When node u wants to send a message to node v, $\mathrm{TS}_{\mathrm{dppu}}$ consists of three phases: first u sends the message to all members of its fan-out set; each member then sends the message to its connected (via a path) pair in v's fan-in set; and finally each member in v's set forwards the message to v, who accepts the value resulting from the majority of received values.

Given such a scheme, our construction proceeds as follows. We start with the technique of forming partially overlapping "committees," introduced by Bracha [6] and used in several other contexts (e.g., [20, 23, 15, 11]). However, while this is a somewhat standard first step, we need to make use of several additional tools in order for the technique to be successful in our a.e. agreement context. Our construction works through the following steps:

1. We create $N = n \log^{k'} n$ committees (for some constant k') such that at most $\mathcal{O}(\frac{N}{\log N})$ of these committees are "bad" (by "bad" here we mean that some threshold fraction of the nodes in the committee are corrupted).
2. For every node in the graph, we assign a poly-logarithmic number of these committees as the *helper committees* for this node. We view these committees as "super-nodes" that can be connected by "super-edges"; when two committees are connected by a super-edge, we have to connect node i in the first committee with node i in the second, ordering nodes in each committee lexicographically.
3. Now, if we assume that these N committees are nodes connected by edges as per the graph G_{dppu} from [10], then using the transmission scheme $\mathrm{TS}_{\mathrm{dppu}}$, one can obtain a large set of committees (call them the "privileged" committees)

such that any two committees can reliably communicate between themselves. This is because only $\mathcal{O}(\frac{N}{\log N})$ of the committees are corrupt. But, for this to work, committees need to be able to communicate across super-edges in exactly the same way as nodes communicate across edges. In order to achieve this task, we will make use of a suitable *differential agreement* protocol, specified in Section 2.3. The validity condition of this agreement protocol has the property that if many honest nodes begin the protocol with some value v then at the end of the protocol all honest nodes will output v. In this way, reliable message transmission across a super-edge can be obtained by having nodes in one committee send the message to nodes in the other committee and running the above differential agreement protocol amongst the nodes of the second committee.
4. We next show that for $n-t-\mathcal{O}(\frac{t}{\log n})$ nodes (those are the "privileged" nodes), most helper committees assigned to these nodes are also privileged committees.
5. The idea now is to first make a privileged sender node u_s send the message m to all its helper committees (most of which are privileged). Second, these helper committees can communicate with the privileged committees in the helper-committee set of the receiver node u_r using the protocol $\mathrm{TS}_{\mathrm{dppu}}$. Finally, u_r can obtain the value m from its set of helper committees.

Organization of the paper. We begin in Section 2 with the description of our model, as well as the definitions and building blocks needed for our protocol. Section 3 is dedicated to our main result: the construction of our poly-logarithmic degree graph, and presentation of our protocol for reliable remote message transmission. Due to space limitations, some of the background material, the "reduction" of a.e. agreement and secure computation on sparse networks to our reliable remote message transmission protocol, as well as proofs, are given in the full version [7] of this paper.

2 Model, Definitions and Building Blocks

Let $G = (V, E)$ denote a graph with n nodes (i.e., $|V| = n$). We also refer to the nodes of the network as parties. The edges of the graph model communication links or channels. We assume a synchronous network and assume that the communication is divided into rounds. In every round, a player can send (possibly different) messages on its incident edges; these messages are delivered before the next round. An adversary $\mathcal{A}$ can corrupt a set of nodes $\mathcal{T}$ in the network such that $\mathcal{T} \subset V, |\mathcal{T}| \leq t$. $\mathcal{A}$ has unlimited computational power, can corrupt nodes adaptively (i.e., can use information obtained from a set of corrupted parties at one round to determine which node(s) to corrupt next) and is *rushing* (i.e., $\mathcal{A}$ can learn messages sent by honest parties before deciding the messages to be sent by corrupted parties in a particular round).

2.1 Expander graphs

Expander graphs are graphs with the property that for any (not too large) subset of nodes S, the number of outgoing edges from this set is proportional to its size. Expander graphs can also be viewed as bipartite graphs. When we wish to consider this representation, we will use the following definition.

Definition 1 *A* bipartite multi-graph *with n left vertices and m right vertices, where every left vertex has degree d, can be specified by a function $\Gamma : [n] \times [d] \rightarrow [m]$, where $\Gamma(u, r)$ denotes the r^{th} neighbor of vertex $u \in [n]$. For a set $S \subseteq [n]$, we write $\Gamma(S)$ to denote the set of neighbors of S. That is, $\Gamma(S) = \{\Gamma(x, y) : x \in S, y \in [d]\}$.*

Definition 2 *A bipartite graph $\Gamma : [n] \times [d] \rightarrow [m]$ is an (n, m, d, l, A)-*expander*, if for every set $S \subseteq [n]$, with $|S| = l$, we have $|\Gamma(S)| \geq A \cdot l$. Γ is an $(n, m, d, \leq l_{max}, A)$ expander if it is an (n, m, d, l, A) expander for every $l \leq l_{max}$.*

A bipartite expander is *balanced* if $m = n$. It is *right-regular* if all nodes on the right also have the same degree D (nodes on the left all have degree d). We will make use of constant-degree balanced expander graphs[4] that are right D-regular, where $A = \epsilon' d$ for some constant $\epsilon', l_{\max} \leq \theta n$ for constant θ and $D = \mathcal{O}(d)$. Such graphs can be constructed using any constant-degree unbalanced ($m < n$) expander following the work of Guruswami *et al.* [14].

2.2 Almost-everywhere agreement

The problem of *almost-everywhere agreement* (a.e. agreement for short) was introduced by Dwork *et al.* [10]. A.e. agreement "gives up" certain honest nodes in the network, which is unavoidable due to their poor connectivity with other honest nodes. We refer to the given-up nodes as *doomed* nodes; the honest nodes for which we guarantee agreement are referred to as *privileged* nodes. Let the set of doomed nodes be denoted by $\mathcal{X}$ and the set of privileged nodes by $\mathcal{P}$. Note that the sets $\mathcal{P}$ and $\mathcal{X}$ are a function of the set of corrupted nodes ($\mathcal{T}$) and the underlying graph. Let $|\mathcal{X}| = x$ and $|\mathcal{P}| = p$. Clearly, we have $p + x + t = n$. We begin with the formal definition of a.e. agreement.

Definition 3 *A protocol for parties $\{P_1, P_2, \cdots, P_n\}$, each holding initial value v_i, is an* almost-everywhere agreement *protocol if the following conditions hold for any adversary $\mathcal{A}$ that corrupts a set of nodes $\mathcal{T}$ with $|\mathcal{T}| \leq t$:*

AGREEMENT: *All nodes in $\mathcal{P}$ output the same value.*
VALIDITY: *If for all nodes in $\mathcal{P}$, $v_i = v$, then all nodes in $\mathcal{P}$ output v.*

The difference with respect to standard Byzantine agreement is that in the latter the two conditions above are enforced on *all* honest nodes, as opposed to only the nodes in $\mathcal{P}$. For brevity, we keep the same names.

In the context of a.e. agreement, one would like the graph G to have as small a degree as possible (i.e., in relation to the size of the graph and to the number of

[4] Strictly speaking, we do not require the expander to be balanced, only that $n = m \log^s m$, for some constant $s \geq 0$.

corrupted parties). We would like the protocol to guarantee agreement amongst $n - \mathcal{O}(t)$ parties, while allowing $t = \alpha n$ for some constant $0 < \alpha < 1$.

Dwork *et al.* constructed graphs with *constant degree* tolerating at most $t = \mathcal{O}(\frac{n}{\log n})$ corruptions and at the same time guaranteeing agreement amongst $n - \mathcal{O}(t)$ nodes in the network. That is, the graph $G_{\text{dppu}} = (V_{\text{dppu}}, E_{\text{dppu}})$ on n nodes has constant degree. The number of corrupted parties t, tolerated by the protocol can be at most $\mathcal{O}(\frac{n}{\log n})$, while the number of doomed nodes is a constant times t. The idea behind the protocol is to simulate a complete graph on the set of privileged nodes. The theorem from [10] is as follows:

Theorem 1 *There exist constants μ and d independent of n and t, an n-vertex d-regular graph G_{dppu} that can be explicitly constructed, and a communication protocol* TS_{dppu} *such that for any set of adversarial nodes $\mathcal{T}$ in G_{dppu} such that $|\mathcal{T}| = t = \mathcal{O}(\frac{n}{\log n})$, the communication protocol guarantees reliable communication between all pairs of nodes in a set of honest nodes $\mathcal{P}$ of size $\geq n - \mu t$, for constant $\mu > 1$. The protocol generates polynomial (in n) number of messages and has polynomial (in n) running time.*

Given the above theorem, Dwork *et al.* observe that one can run any Byzantine agreement protocol designed for a fully connected graph on G_{dppu} by simulating all communication between nodes in the network with the communication protocol TS_{dppu}. We remark that this results in a slow down of the agreement protocol by a factor proportional to the diameter of the graph. (A high-level idea of how TS_{dppu} works was given in the overview of our techniques in Section 1; we refer the reader to [10] for further details.)

As a result, let μ, d, and t be as defined above and let $\text{BA}(n, t')$ be an agreement protocol for a complete network with up to $t' = \mu t$ faulty processors. Then, simulating the protocol $\text{BA}(n, t')$ on the network G_{dppu} using the communication protocol TS_{dppu}, guarantees agreement among at least $n - \mu t$ honest nodes in the presence of up to $t = \mathcal{O}(\frac{n}{\log n})$ faulty nodes.

2.3 Differential agreement

Fitzi and Garay [12] introduced the problem of *δ-differential agreement* (also, "consensus") developing on the so-called "strong consensus" problem [19], in which every party begins with an input v from a larger (than binary) domain $\mathcal{D}$.[5] We describe the problem below and state the results from [12].

In the standard Byzantine agreement problem, n parties attempt to reach agreement on some value v (either 0 or 1). Let c_v denote the number of honest parties whose initial value is v, and δ be a non-negative integer. δ-differential agreement is defined as follows:

Definition 4 *A protocol for parties $\{P_1, P_2, \cdots, P_n\}$, each holding initial value v_i, is a δ-*differential agreement *protocol if the following conditions hold for any adversary $\mathcal{A}$ that corrupts a set $\mathcal{T}$ of parties with $|\mathcal{T}| \leq t$:*

[5] In contrast to standard Byzantine agreement, the validity condition in the strong consensus problem states that the output value v must have been the input of some honest party P_i (which is implicit in the case of binary Byzantine agreement)

AGREEMENT: *All honest parties output the same value.*
δ-DIFFERENTIAL VALIDITY: *If the honest parties output v, then $c_v + \delta \geq c_{\bar{v}}$.*

Theorem 2 *[12] In a synchronous, fully connected network, δ-differential agreement is impossible if $n \leq 3t$ or $\delta < t$. On the other hand, there exists an efficient (i.e., polynomial-time) protocol that achieves t-differential agreement for $n > 3t$ in $t+1$ rounds.*

Let $\mathrm{DA}(n, t, \delta)$ denote a δ-differential agreement protocol for a fully connected network tolerating up to t faulty processors.

3 Reliable Remote Communication on Sparse Networks

In this section we show how to build a reliable message transmission scheme, $\mathrm{TS}_{\mathrm{low}}$, between any two nodes u_i and u_j that are in a large set of privileged nodes $\mathcal{P}$, on a low-degree (i.e., poly-logarithmic) graph. We begin with the construction of the graph, followed by the description of the transmission scheme, followed by its proof of correctness.

Given only poly-logarithmic degree and a linear number of corruptions, constructing a reliable remote message transmission scheme is significantly harder than in the case of [10]. As mentioned before, the technique used by Upfal [22] of simply "flooding" all the paths with the message, and showing that at least one corrupted path exists, leads to an exponential-time algorithm. To avoid that, we will need to obtain multiple good paths between two privileged nodes using just poly-logarithmic degree, even though a linear number of nodes may be corrupted. A salient feature of our transmission scheme compared to those of [10] and [22] will be that nodes along the path in our scheme play and active role and do not just simply forward the message being sent. This is done in order to ensure that the correct message is transmitted along a majority of the paths.

3.1 The low-degree graph construction

The graph $G = (V, E)$ that we construct is as follows. Let the nodes in V be denoted by $u_1, u_2, \cdots, u_n$. Let $G_{\mathrm{exp}} = (V, E_{\mathrm{exp}})$ be an (n, d, ϵ)-expander graph for constants $d, \epsilon > 0$ on the nodes $u_1, u_2, \cdots, u_n$. We begin, by first forming partial overlapping "virtual committees." The notion of such committees was introduced by Bracha [6] in the context of Byzantine agreement and has since been used in the construction of several protocols for other problems, including leader election, secure message transmission and secure multi-party computation [20, 23, 15, 11].

We form committees in the following manner. Start at any node $u_i \in V$, and consider all possible walks of length $\gamma = k \log\log n$ starting at this node. Group nodes in each walk to form a committee C_j. We repeat this procedure beginning at every node in V. Let $k' = k \log d$. Note that, by the above procedure, we create $N = n \log^{k'} n$ committees $C = \{C_1, \cdots, C_N\}$, each of size $\gamma = k \log\log n$. We construct the set of edges E in graph G in the following three ways:

1. First, If $u_i, u_j \in C_l$ for some C_l, then we connect u_i and u_j by an edge. In other words, we connect all nodes within a committee by a clique.
2. Next, let $G_{\text{dppu}} = (V_{\text{dppu}}, E_{\text{dppu}})$ be a constant-degree r-regular graph on the "nodes" $C = \{C_1, C_2, \cdots, C_N\}$ as constructed in the work of [10]. Now, if $(C_i, C_j) \in E_{\text{dppu}}$, then for all $1 \leq l \leq \gamma$, we connect the l^{th} node in C_i with the l^{th} node in C_j, ordering nodes in C_i and C_j lexicographically; we say "C_i and C_j are connected by a super-edge" to express this.
3. Finally, let G_{biexp} be a $(N, N, d', \leq \theta N, \epsilon' d')$ bipartite expander graph that is right D-regular for constants $d', \epsilon', \theta, D > 0$. Let the nodes on the left of this graph (call it V_l) represent the nodes of G (i.e., V) where each node $u_i \in V$ appears in V_l a $(\log^{k'} n)$ number of times; we use $u_{i,1}, \cdots, u_{i,\log^{k'} n}$ to denote these $\log^{k'} n$ nodes. Let the nodes on the right of this graph (call it V_r) represent the N committees formed by the above outlined method. Now, if $(u_{i,m}, C_j)$, $1 \leq m \leq \log^{k'} n$, is an edge in G_{biexp}, we connect u_i and all nodes in C_j by an edge.

This completes the construction of graph $G = (V, E)$. In the full version [7], we show that the degree of every node in G is poly-logarithmic in n.

3.2 The reliable remote message transmission scheme

We begin with a high-level description of our reliable remote message transmission scheme, TS_{low}.

High-level intuition. Let the number of nodes in the network be n and let the number of corrupt nodes be $t = \alpha n$. Using the n nodes of G, following our graph construction above we formed $N = n \log^{k'} n$ committees each of size $k \log \log n$. We call a committee *bad* if the number of corrupted nodes in it is $\geq \frac{k \log \log n}{4}$; otherwise, we call a committee *good*. Now, again according to our graph construction, nodes within each committee are connected by a clique and hence can run any standard Byzantine agreement protocol successfully within themselves if the number of corrupted nodes in them is less than $\frac{1}{4}^{\text{th}}$ fraction. In other words, any honest node in a good committee can agree upon a value with other honest nodes in the committee.

We begin by showing that the number of bad committees (or super-nodes) is at most $T = \frac{N}{c\mu \log N}$, for constants $\mu, c > 1$. Recall that the N committees are connected by the graph G_{dppu}. Therefore, out of these N committees, at least $N - \mu T$ of them are "privileged" and hence any two privileged committees can execute reliable remote message transmission between themselves.

In order to do this, we will describe a protocol for simulating the transmission of message m across a super-edge that connects two good committees. In particular, we need to make sure that while transmitting a message across a path consisting of committees, the number of honest players holding the message m in every committee along the path, remains the same. For this, we make use of a differential agreement protocol (Section 2.3).

Now that we have the guarantee that any two privileged committees can reliably communicate, we can outline our transmission scheme. Recall that every node u_i is connected to $\beta = d' \log^{k'} n$ committees according to the bipartite expander graph G_{biexp}. Let these β committees be called *helpers* (or *helper committees*) of node u_i. We will show that for at least $n - \frac{\mu' t}{\log n}$ (for some constant μ') of the nodes in V (this will be the set of privileged nodes $\mathcal{P}$), more than $\frac{5\beta}{6}$ of the nodes' helpers are privileged committees. This means that for every privileged node u_i, more than a $\frac{5}{6}^{\text{th}}$ fraction of u_i's helper committees are privileged according to the [10] graph connecting the committees.

Suppose now that $u_i \in \mathcal{P}$ wishes to send message m reliably to $u_j \in \mathcal{P}$. u_i first sends m to all its helper committees; call these helper committees $C_1^{u_i}, \cdots, C_\beta^{u_i}$. Let u_j's helper committees be denoted by $C_1^{u_j}, \cdots, C_\beta^{u_j}$. Next, for all $1 \leq l \leq \beta$, $C_l^{u_i}$ sends message m to $C_l^{u_j}$ using the transmission scheme TS_{dppu} simulating the communication across super-edges with the (differential agreement-based) protocol mentioned before. Since we have the guarantee that more than a $\frac{5}{6}^{\text{th}}$ fraction of u_i and u_j's helper committees are privileged and any two privileged committees can reliably send messages to each other (according to [10]), we have that a $\frac{2}{3}^{\text{rds}}$ majority ($> \frac{1}{2}$) of the helper committees of u_j receive the message m. Since in a privileged committee, greater than a $\frac{3}{4}^{\text{ths}}$ fraction of the nodes are honest, a simple majority ($> \frac{2}{3} \times \frac{3}{4}$) of the nodes (these nodes are honest) in the helper committees of u_j receive message m. Finally, u_i simply receives a value from all its helper committees and takes a majority of the values received. We now describe the transmission scheme in detail.

Message transmission between committees. We begin by describing the protocol for reliable remote message transmission executed by committees. We have a set of N committees, $\mathcal{C} = \{C_1, \cdots, C_N\}$, each of size $\gamma = k \log\log n$. Nodes within a committee are connected by a clique. The N committees are connected through super-edges according to a constant degree graph G_{dppu} from [10]. A super-edge between two committees C_i and C_j is obtained by connecting the l^{th} node in C_i to the l^{th} node in C_j (for all $1 \leq l \leq \gamma$), ordering the nodes in C_i and C_j lexicographically. A committee is *bad* if at least $\frac{k \log\log n}{4}$ of the nodes in it are corrupt. Let the number of bad committees be denoted by $\mathcal{T}_C$, with $|\mathcal{T}_C| = T$. We have a sender committee C_s and a receiver committee C_r. Both these committees are good and furthermore are "privileged" according to the graph G_{dppu}. All honest nodes in C_s begin the protocol with a message m. We require that at the end of the protocol all honest nodes in C_r output the same message m. Furthermore, we require that the set of privileged committees $\mathcal{P}_C$ be large; i.e., at least $N - \mu T$ for some constant $\mu > 0$. We outline such a protocol below, which essentially consists of running the TS_{dppu} transmission scheme, but at a committee level:

$\text{TS}^{\mathcal{C}}_{\text{dppu}}(\mathbf{C_s}, \mathbf{C_r}, \mathbf{T}, \mu, \mathbf{m})$:

1. Committees $\{C_1, \cdots, C_N\}$ view themselves as nodes in the graph G_{dppu}. C_s uses the reliable remote message transmission scheme TS_{dppu} to send mes-

sage m to C_r. However, whenever committee C_i is supposed to send message m to committee C_j, such that C_i and C_j are connected by a super-edge, the committees execute the protocol $\mathrm{Send}^{\mathcal{C}}(C_i, C_j, m)$ described below.

2. Let the output of committee C_r be m_r from the above protocol. Every node $u \in C_r$ outputs m_r as the output of the protocol.

$\mathrm{Send}^{\mathcal{C}}(\mathbf{C_i}, \mathbf{C_j}, \mathbf{m})$:

1. Let the nodes in C_i be denoted by $u_1^i, \cdots, u_\gamma^i$ and the nodes in C_j be denoted by $u_1^j, \cdots, u_\gamma^j$. u_l^i sends message m to u_l^j across the edge connecting the two nodes, for all $1 \le l \le \gamma$. Let the value received by u_l^j be denoted by m_l^j.
2. The nodes $u_1^j, \cdots, u_\gamma^j$ execute a differential agreement protocol $\mathrm{DA}(\gamma, \lceil \frac{\gamma}{4} \rceil - 1, \lceil \frac{\gamma}{4} \rceil - 1)$ with inputs $m_1^j, \cdots, m_\gamma^j$. Let the value agreed by honest nodes in C_j be denoted by m'. The output of the committee C_j in the protocol is m'.

The main protocol. Let $u_i, u_j \in V$ be two privileged nodes. u_i is connected to $\beta = d' \log^{k'} n$ helper committees $C_1^{u_i}, \cdots, C_\beta^{u_i}$. In turn, let the nodes in $C_l^{u_i}$ be denoted by $C_l^{u_i}[1], \cdots, C_l^{u_i}[\gamma]$, where $\gamma = k \log\log n$ (likewise for u_j). The protocol for reliable remote message transmission between two privileged nodes u_i and u_j is given below.

$\mathrm{TS}_{\mathrm{low}}(\mathbf{u_i}, \mathbf{u_j}, \mathbf{m})$:

1. For all $1 \le l \le \beta, 1 \le w \le \gamma$, u_i sends $C_l^{u_i}[w]$ the value m using the edge connecting u_i and $C_l^{u_i}[w]$.
2. For all $1 \le l \le \beta$, $C_l^{u_i}$ sends $C_l^{u_j}$ message m using protocol $\mathrm{TS}_{\mathrm{dppu}}^{\mathcal{C}}(C_l^{u_i}, C_l^{u_j}, T, \mu, m)$.
3. For all $1 \le l \le \beta, 1 \le w \le \gamma$, let $m_l^{u_j}[w]$ denote the output of node $C_l^{u_j}[w]$ after execution of protocol $\mathrm{TS}_{\mathrm{dppu}}^{\mathcal{C}}(C_l^{u_i}, C_l^{u_j}, T, \mu, m)$. For all $1 \le l \le \beta, 1 \le w \le \gamma$, $C_l^{u_j}[w]$ sends node u_j the value $m_l^{u_j}[w]$ using the edge connecting $C_l^{u_j}[w]$ and u_j.
4. u_j takes the majority of all $m_l^{u_j}[w]$ values received and outputs this majority (call it m') as the output of the protocol.

3.3 Proof of correctness

We prove the correctness of our transmission scheme (theorem below) through a series of lemmas that can be found in the full version [7] of this paper.

Theorem 3 *Let u_i and u_j be two privileged nodes in graph G. Then $\mathrm{TS}_{low}(u_i, u_j, m)$ is a reliable remote message transmission protocol between u_i and u_j.*

To conclude, following the approaches of [10, 13], we show in the full version [7] how to use our transmission scheme to achieve a.e. agreement on graph G, and a.e. secure computation on a graph with degree a constant times G's.

References

1. A. Bagchi, A. Bhargava, A. Chaudhary, D. Eppstein, and C. Scheideler. The effect of faults on network expansion. *Theory Comput. Syst.*, 39(6):903–928, 2006.
2. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC'88*, pages 1–10, 1988.
3. M. Ben-Or and D. Ron. Agreement in the presence of faults, on networks of bounded degree. *Information Processing Letters*, 57:329–334, 1996.
4. P. Berman and J. Garay. Asymptotically optimal distributed consensus (extended abstract). In *ICALP'89*, pages 80–94, 1989.
5. P. Berman and J. Garay. Fast consensus in networks of bounded degree (extended abstract). In *WDAG'90*, pages 321–333, 1990.
6. G. Bracha. An $O(\log n)$ expected rounds randomized Byzantine generals protocol. In *STOC'85*, pages 316–326, 1985.
7. N. Chandran, J. Garay, and R. Ostrovsky. Improved fault tolerance and secure computation on sparse networks. CoRR, 2010. http://arxiv.org/.
8. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (abstract). In *STOC'88*, pages 11–19, 1988.
9. D. Dolev and R. Reischuk. Bounds on information exchange for byzantine agreement. In *PODC'82*, pages 132–140, 1982.
10. C. Dwork, D. Peleg, N. Pippenger, and E. Upfal. Fault tolerance in networks of bounded degree (preliminary version). In *STOC'86*, pages 370–379, 1986.
11. M. Fitzi, M. Franklin, J. Garay, and S. Vardhan. Towards optimal and efficient perfectly secure message transmission. In *TCC'07*, pages 311–322, 2007.
12. M. Fitzi and J. Garay. Efficient player-optimal protocols for strong and differential consensus. In *PODC'03*, pages 211–220, 2003.
13. J. Garay and R. Ostrovsky. Almost-everywhere secure computation. In *EUROCRYPT'08*, pages 307–323, 2008.
14. V. Guruswami, J. Lee, and A. Razborov. Almost euclidean subspaces of ℓ_1^N via expander codes. In *SODA'08*, pages 353–362, 2008.
15. M. Hirt and U. Maurer. Player simulation and general adversary structures in perfect multiparty computation. *Journal of Cryptology*, 13(1):31–60, 2000.
16. V. King and J. Saia. From almost everywhere to everywhere: Byzantine agreement with $\tilde{\mathrm{O}}(\mathrm{n}^{3/2})$ bits. In *DISC'09*, pages 464–478, 2009.
17. V. King, J. Saia, V. Sanwalani, and E. Vee. Towards secure and scalable computation in peer-to-peer networks. In *FOCS'06*, pages 87–98, 2006.
18. L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.
19. G. Neiger. Distributed consensus revisited. *Information Processing Letters*, 49(4):195–201, 1994.
20. R. Ostrovsky, S. Rajagopalan, and U. Vazirani. Simple and efficient leader election in the full information model. In *STOC'94*, pages 234–242, 1994.
21. M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27:228–234, 1980.
22. E. Upfal. Tolerating linear number of faults in networks of bounded degree. In *PODC'92*, pages 83–89, 1992.
23. D. Zuckerman. Randomness-optimal sampling, extractors, and constructive leader election. In *STOC'96*, pages 286–295, 1996.