1-1-1988

Fault Tolerance in Networks of Bounded Degree

Cynthia Dwork
IBM Almaden Research Center

David Peleg
IBM Almaden Research Center

Nicholas Pippenger
Harvey Mudd College

Eli Upfal
IBM Almaden Research Center

SIAM J. COMPUT.
Vol. 17, No. 5, October 1988

009

FAULT TOLERANCE IN NETWORKS OF BOUNDED DEGREE*

CYNTHIA DWORK†, DAVID PELEG†, NICHOLAS PIPPENGER†, AND ELI UPFAL†

Abstract. Achieving processor cooperation in the presence of faults is a major problem in distributed systems. Popular paradigms such as Byzantine agreement have been studied principally in the context of a complete network. Indeed, Dolev [*J. Algorithms*, 3 (1982), pp. 14–30] and Hadzilacos [*Issues of Fault Tolerance in Concurrent Computations*, Ph.D. thesis, Harvard University, Cambridge, MA, 1984] have shown that $\Omega(t)$ connectivity is necessary if the requirement is that *all* nonfaulty processors decide unanimously, where t is the number of faults to be tolerated. We believe that in forseeable technologies the number of faults will grow with the size of the network while the degree will remain practically fixed. We therefore raise the question whether it is possible to avoid the connectivity requirements by slightly lowering our expectations. In many practical situations we may be willing to "lose" some correct processors and settle for cooperation between the vast majority of the processors. Thus motivated, we present a general simulation technique by which vertices (processors) in almost any network of bounded degree can simulate an algorithm designed for the complete network. The simulation has the property that although some correct processors may be cut off from the majority of the network by faulty processors, the vast majority of the correct processors will be able to communicate among themselves undisturbed by the (arbitrary) behavior of the faulty nodes.

We define a new paradigm for distributed computing, almost-everywhere agreement, in which we require only that almost all correct processors reach consensus. Unlike the traditional Byzantine agreement problem, almost-everywhere agreement can be solved on networks of bounded degree. Specifically, we can simulate any sufficiently resilient Byzantine agreement algorithm on a network of bounded degree using our communication scheme described above. Although we "lose" some correct processors, effectively treating them as faulty, the vast majority of correct processors decide on a common value.

Key words. fault tolerance, communication, bounded-degree network, expander graph

AMS(MOS) subject classifications. 68M10, 68M15, 68R10

1. Preliminaries.

In 1982 Dolev [D] published the following damning result for distributed computing: "Byzantine agreement is achievable only if the number of faulty processors in the system is less than one-half of the connectivity of the system's network." Even in the absence of malicious failures connectivity $t+1$ is required to achieve agreement in the presence of t faulty processors [H].

The results are viewed as damning because of the fundamental nature of the Byzantine agreement problem. In this problem each processor begins with an initial value drawn from some domain V of possible values. At some point during the computation, during which processors repeatedly exchange messages and perform local computations, each processor must irreversibly decide on a value, subject to two conditions. No two correct processors may decide on different values, and if all correct processors begin with the same value v, then v must be the common decision value. (See [F] for a survey of related problems.) The ability to achieve this type of coordination is important in a wide range of applications, such as database management, fault-tolerant analysis of sensor readings, and coordinated control of multiple agents.

A simple corollary of the results of Dolev and Hadzilacos is that in order for a system to be able to reach Byzantine agreement in the presence of up to t faulty processors, every processor must be directly connected to at least $\Omega(t)$ others. Such high connectivity, while feasible in a small system, cannot be implemented at reasonable cost in a large system.

As technology improves, increasingly large distributed systems and parallel computers will be constructed. However, in any forthcoming technology, the number of

* Received by the editors June 17, 1986; accepted for publication (in revised form) November 3, 1987.
† IBM Almaden Research Center, San Jose, California 95120-6099.

faulty processors in a given system will grow with the size of the system, whereas the degree of the interconnection network by which the processors communicate will, for all practical purposes, remain fixed.

Despite these negative results, distributed systems are widely used and parallel computers are being built. This suggests that the correctness conditions for Byzantine agreement are too stringent to reflect practical situations. In particular, Byzantine agreement guarantees coordination among all correct processors, by necessarily omitting up to t faulty processors. In many situations it may suffice to guarantee agreement among all but $O(t)$ processors. In other situations a simple majority consensus may suffice. Similarly, in clock synchronization, or in firing squad synchronization, it may suffice for a vast majority of the correct processors to be synchronized.

In the traditional paradigm for distributed computing described above, the correctness conditions describe the states of *all* nonfaulty processors. In this paper we propose a new paradigm for fault-tolerant computing in which correctness conditions are relaxed by "giving up for lost" those correct processors whose communication paths to the remainder of the network are excessively corrupted by faulty processors. Such a processor is called *poor*. While any network of bounded degree must contain some poor processors, in this paper we show that their number can often be kept quite small, even in networks of constant degree. Further, we argue that this type of cooperation may fit well most applications of, say, Byzantine agreement. All known algorithms guarantee only that if at most $f \leqq t < n/3$ processors fail then at least $k \geqq n - f$ processors will mutually agree on a value. Our results show that we can eliminate the costly connectivity condition requiring $\Omega(nt)$ edges by employing an appropriately chosen bounded-degree network of $n + O(t)$ processors and still guarantee agreement among n correct processors. Our paradigm admits deterministic solutions in networks of small constant degree to such fundamental problems as atomic broadcast, Byzantine agreement, and clock synchronization.

We present a general simulation technique by which for almost any regular graph G, the vertices (processors) of G can simulate an algorithm designed for a complete network in such a way that the number of poor processors in G is small. The crux of the simulation is a transmission scheme for simulating the point-to-point transmissions of the complete network by sending messages along several paths of G in such a way that there will always be a large set of correct processors capable of communicating among themselves as if they comprise a fully connected subnetwork, independent of the behavior of the faulty processors.

For consensus problems we can often do better than in the general simulation by employing a *compression procedure* based on the existence of *compressor graphs* [P]. This procedure is iterative and local in nature, and cannot by itself guarantee agreement. However, it can be used to "sharpen" dichotomies in that if a sufficiently large majority (e.g., all but $O(t \log t)$) of the correct processors have the same value, then the procedure converges and strengthens this majority (e.g., to all but $t+1$).

Our model of computation is identical to that commonly used in the Byzantine literature. Specifically, each processor can be thought of as a (possibly infinite) state machine with special registers for communication with the outside world. The processors communicate by means of point-to-point links, which are assumed to be completely reliable. The entire system is synchronous, and can be thought of as controlled by a common clock. At each pulse of the common clock a processor may send a message on each of its incident communication links (possibly different messages on different links). Messages sent at one clock pulse are delivered before the next pulse.

For each of our transmission schemes there is a specific lower bound b on the number of clock pulses needed to simulate one complete round of message exchange

in the simulated network. For simplicity we assume that the common clock sends a "super-pulse" every b rounds. A processor simulates round r of the original algorithm at the rth super-pulse.

Since we cannot hope to solve the Byzantine agreement problem exactly on networks of bounded degree, we introduce the notion of *almost-everywhere agreement* (denoted a.e. agreement), in which all but a small number of the correct processors must choose a common decision value.

More precisely, a protocol P is said to achieve t-resilient X *agreement*, where X is any term, if in every execution of P in which at most t processors fail all but X of the correct processors eventually decide on a common value. Moreover, if all the correct processors share the same initial value then that must be the value chosen. Note that the traditional Byzantine agreement problem is just 0 agreement.

A protocol solves a.e. agreement if it solves X agreement for some X such that $X/(n-t)\to 0$ as $n\to\infty$.

Our first result applies only to fail stop, omission, or authenticated Byzantine faults.

THEOREM 1. *For all $r\geqq 5$ there exists a constant $\varepsilon=\varepsilon(r)$ such that for all $t<\varepsilon n$ almost all r-regular graphs (i.e., all but a vanishingly small fraction of such graphs) admit a t-resilient algorithm for $O(t)$ agreement.*

The remaining results apply to unauthenticated Byzantine failures.

THEOREM 2. *For all $r\geqq 5$, almost all r-regular graphs admit a t-resilient algorithm for $O(t)$ agreement, where $t\leqq n^{1-\varepsilon}$, for some constant $\varepsilon=\varepsilon(r)$, where $\varepsilon(r)\to 0$ as $r\to n$.*

The next theorems describe explicit graphs for which the set of poor processors is small.

THEOREM 3. *The n node butterfly network (degree 4; see § 2.3 for definition) admits a t-resilient $O(t\log t)$-agreement algorithm for $t\leqq cn/\log n$ for some constant c.*

The result of Theorem 3 can be improved for a family of networks obtained by superimposing a compressor of degree 5 on a butterfly network.

THEOREM 4. *There exists a constant c and a network of degree 9 that admits a t-resilient $O(t)$-agreement algorithm for $t\leqq cn/\log n$.*

In the case of unauthenticated Byzantine failures, we achieve $O(t)$ agreement only for $t\leqq cn/\log n$. If $t>O(n/\log n)$ then it is easy to show that the number of poor processors is linear in n. The existence problem for an $O(t)$-agreement algorithm in this case remains open. However, we solve this problem on graphs of unbounded but still relatively small degree.

THEOREM 5. *For every $0<\varepsilon<1$ there exist a constant $c=c(\varepsilon)$, graphs G of degree $O(n^{\varepsilon})$, and t-resilient $O(t)$-agreement algorithms for $t\leqq cn$.*

Finally we present a purely combinatorial characterization of networks which admit $p(t)$ agreement for any function p. When $p(t)=0$ our characterization coincides exactly with the $(2t+1)$-connectivity requirement for the traditional Byzantine agreement cited above [D].

2. Simulation results. In § 2.1 we describe a general strategy for simulating on one network any algorithm designed for another network, describing what we mean by "simulation." In § 2.2 we discuss a general scheme for implementing our strategy, and in § 2.3 we make all of this more concrete by presenting the simulation of a complete network by a butterfly network. In § 2.4 we show that our general scheme can be implemented on almost all regular graphs of bounded degree. Finally, § 2.5 briefly discusses our results under more restrictive fault models.

2.1. The general simulation. For simplicity, we take the simulated network to be completely connected. Let A be an algorithm designed for a fully connected network H. Consider an arbitrary network G over the same set of vertices (processors) as in

H, and suppose we wish to simulate A on G. We need only specify the simulation of communication between processors; a direct message from a processor u to its neighbor v in H can be simulated in G by sending the message from u to v through various paths, and supplying v with a method for determining the correct value of the message, e.g., by taking the value appearing in the majority of the paths. Taken together, the particular choice of paths and the supplied decision method constitute a *transmission scheme.* Of course, even if u and v are correct processors the faulty processors may be so placed that all or most of the paths from u to v are corrupted. Thus, even if a processor is correct, it may be unable to properly communicate with the other processors.

Let a transmission scheme for G be fixed. Let T be a subset of the vertices of G (think of T as the set of faulty processors). A pair of nodes $(u, v) \in G$ is *successful* with respect to T if, whenever all the processors not in T follow the transmission scheme correctly, the simulation of a message transmission from u to v always succeeds (i.e., v decides correctly on the value sent by u). Let POOR (G, T) be a minimal set of correct nodes such that every pair of nodes, $u, v \notin T \cup \text{POOR}(G, T)$ is successful with respect to T. (Note that this set need not be unique.) Let $p(G, t) = \max\{|\text{POOR}(G, T)| \text{ such that } T \subseteq V, |T| = t\}$. As we will show in Theorem 2.1 there is a $p(G, t)$-agreement algorithm resilient to t failures for every graph G and suitable choice of t. We are therefore interested in finding graphs G for which $p(G, t)$ is small. Such graphs are the subject of §§ 2.3 and 2.4.

THEOREM 2.1. *Let A be an algorithm for the traditional Byzantine agreement problem designed for network H, let G be a graph with the same number of vertices as H, and let TS be a transmission scheme for simulating on G message transmissions in H. Let $A(TS)$ be the simulation of A on G using the transmission scheme TS to simulate messages sent on H. For every t, if A is guaranteed to work correctly on H in the presence of at most $t + p(G, t)$ faults, then $A(TS)$ achieves $p(G, t)$ agreement on G in the presence of up to t faults.*

Proof. Let T be a set of faulty processors in G. $A(TS)$ simulates the execution of A on H by simulating the processors in a one-one fashion. By definition, the processors not in POOR (G, T) can communicate among themselves as if they comprised a fully connected subnetwork, so the simulated communication among this set of processors is successful. The behavior of the correct processors in POOR (G, T) may appear to be faulty. These are the processors we give up for lost. Since A is guaranteed to work correctly even in the presence of $t + p(G, t) \geqq |\text{POOR}(G, T)|$ failures we are done. □

In order to use the transmission schemes described here the processors must have some knowledge of the topology of the system. The amount of knowledge needed, and how this quantity depends on the types of faults considered, are subjects for further research.

2.2. A class of transmission schemes. We now describe in more detail a specific class of transmission schemes, called *three-phase* transmission schemes. Let G be any network in which we may specify the following sets. For every node v we specify sets of processors $\Gamma_{\text{in}}(v)$, $\Gamma_{\text{out}}(v) \subseteq V$, each of fixed (but not necessarily constant) size s. For each node w in $\Gamma_{\text{in}}(v)$ ($\Gamma_{\text{out}}(v)$) a path from w to v (v to w) is specified. In addition, for each ordered pair of nodes (u, v) we specify s vertex-disjoint paths from $\Gamma_{\text{out}}(u)$ to $\Gamma_{\text{in}}(v)$.

The transmission of a message x from u to v consists of three phases. In the first phase the message is broadcast from u to every node in $\Gamma_{\text{out}}(u)$ through the specified paths. Thus, at the end of the first phase a copy of x is received by all nodes of $\Gamma_{\text{out}}(u)$.

(Those processors in $\Gamma_{out}(u)$ whose path from u contains faults may have received an incorrect version of the message, or nothing at all.) In the second phase the s (possibly corrupted) copies of u's message are sent along the vertex-disjoint paths from $\Gamma_{out}(u)$ to $\Gamma_{in}(v)$. In the third phase these (possibly corrupted) copies of u's message are routed to v along the specified paths from $\Gamma_{in}(v)$ to v. Thus, v receives s (possibly corrupted) copies of u's message. Finally, v takes the value appearing in the majority of the copies that arrived to be the actual message. (If no majority exists then a default value is taken.) Clearly, v could be making a mistake, even if it is a correct processor.

Let T be some set of nodes on the network (again, think of T as the set of faulty processors). A node u is said to be *out-bad* with respect to T if at least $\frac{1}{8}$ of the specified paths to $\Gamma_{out}(u)$ contain a vertex in T. Intuitively, if T is the set of faulty processors and u is out-bad with respect to T, then at least $\frac{1}{8}$ of the processors in $\Gamma_{out}(u)$ may receive a corrupted version of the message x. Similarly we say v is *in-bad* with respect to T if at least $\frac{1}{8}$ of the paths from $\Gamma_{in}(v)$ to v pass through some node of T. Let BAD (G, T) be the set of nodes in the network G which are (either out- or in-) bad with respect to T. Let $b(G, t) = \max\{|\text{BAD}(G, T)| \text{ such that } T \subseteq V, |T| = t\}$. Recall that, informally, $p(G, t)$ is an upper bound on the number of poor processors in G when no more than t processors fail. The next claim bounds the number of poor processors in terms of the number of bad processors.

CLAIM 2.2. *Let s be the size of the sets Γ. For all $t < s/4$, $p(G, t) \leqq b(G, t)$.*

Proof. We will prove that for any $T \subseteq V$, POOR $(G, T) \subseteq$ BAD (G, T). The claim then follows immediately from the definitions. Let T be some subset of V of size at most t. We will show that for any two processors u and v not in BAD (G, T), any message m sent from u to v according to the three-phase transmission scheme is correctly received by v. This follows from the following argument: since u is not out-bad, m will reach at least $\frac{7}{8}$ of its first-phase destinations correctly. Thus at the end of the first phase, at most $s/8$ copies of m have been corrupted. In the second phase, since every copy is transmitted along disjoint paths, at most t more copies might get hurt. Finally in the third phase, since v is not in-bad, at most an additional $s/8$ copies can be corrupted. Over all, at most $s/4 + t < s/2$ copies of the message might be lost. Therefore the majority of the copies will arrive intact, so v will recognize the message correctly. □

Remark. It will sometimes be necessary to consider a more relaxed type of three-phase transmission scheme in which each node $v \in (\Gamma_{out}(u) \cup \Gamma_{in}(w))$ may appear on at most two of the paths between these two sets (once as an endpoint and at most once more as an intermediate node). The paths have to remain otherwise disjoint. For such a scheme we can prove that if s is the size of the sets Γ, then for all $t < s/8$, $p(G, t) \leqq b(G, t)$.

2.3. Simulation of a complete network on the butterfly. In this section we show how to simulate a complete network on a simple degree-4 network known as the butterfly [U]. All we have to do is specify a transmission scheme. This is done losing at most $O(t \log t)$ processors in the presence of t faults (i.e., at most $O(t \log t)$ processors will be bad as defined in § 2.2).

An m-butterfly is a communication graph $G_m = (V_m, E_m)$ with $V_m = \{(a, i) | 0 \leqq a \leqq m-1, 0 \leqq i \leqq 2^m - 1\}$. The set of edges E_m connects (a, i) to (b, j) if and only if $b = (a+1) \pmod m$ and j is either identical to i or differs from it in the ath least significant bit.

On the butterfly we use a version of the three-phase scheme for transmitting a message from $u = (a, i)$ to $v = (b, j)$. To do this we need only specify the out- and

in-sets and the three sets of paths. All messages are sent only through forward links (i.e., from a node (c, k) to a node $((c+1)(\text{mod } m), l)$ but not in the other direction). We take $s = 2^m$ and define the sets Γ by $\Gamma_{\text{in}}((c, k)) = \Gamma_{\text{out}}((c, k)) = \{(c, l) | 0 \leqq l \leqq 2^m - 1\}$.

The paths are defined as follows. The forward edges of any node u span a full binary tree of height m whose leaves are all the nodes in $\Gamma_{\text{out}}(u)$, and the paths are chosen according to this tree. The second-phase paths connect every node (a, k) in $\Gamma_{\text{out}}(u)$ to (b, k) in $\Gamma_{\text{in}}(v)$ by $(b-a)(\text{mod } m)$ edges connecting every intermediate node (c, k) to $((c+1)(\text{mod } m), k)$. The third-phase paths from $\Gamma_{\text{in}}(u)$ to u are defined in a way similar to the first-phase paths, using the dual tree based on the backward edges. More specifically, the path from a node (c, k) is directed to its neighbor $((c+1)(\text{mod } m), l)$ with the cth bit of l matching that of j.

This completes the description of a three-phase transmission scheme for the butterfly network. We now analyze the resiliency of this scheme.

CLAIM 2.3. $b(G_m, t) = O(t \log t)$.

Proof. Let us measure the amount of "damage" that a faulty processor p can cause to correct senders in the first phase. Keeping p fixed and looking at all possible senders u whose paths to $\Gamma_{\text{out}}(u)$ contain p we see that p can block at most $1/2^i$ of the outbound paths for its 2^i "distance i" neighbors. Summing up to distance $\log 16t$, the total damage caused by t faulty processors (measured in "number of dominated paths," or "number of destroyed copies") is

$$t\left(2\tfrac{1}{2} + 4\tfrac{1}{4} + \cdots + 2^{\log 16t}\frac{1}{2^{\log 16t}}\right)2^m = t\,(\log 16t)2^m.$$

The number of nodes that might lose $\frac{1}{16}$ or more of their paths due to interruptions of distance $\log 16t$ or less is therefore bounded by $t \log 16t(2^m/(2^m/16)) = 16t \log 16t$. Now, let u be any processor for which less than $\frac{1}{16}$ of the paths from u to $\Gamma_{\text{out}}(u)$ contain a faulty processor at distance $\log 16t$ or less from u. We claim u cannot be out-bad. This would imply that the number of out-bad nodes is also bounded by $16t \log 16t$. To prove the claim, note that if less than $\frac{1}{16}$ of u's out-bound paths are corrupted at distance $\log 16t$ or less, then even if we assume the existence of up to t distinct faulty processes at distance $\log 16t$ or more from u, these faults damage at most $t2^m/2^{\log 16t} = 2^m/16$ of the out-bound paths from u, so less than $\frac{1}{8}$ of the paths from u to $\Gamma_{\text{out}}(u)$ are corrupted, and by definition u is not out-bad. The analysis for the in-bad nodes is identical. Thus, $b(G_m, t) \leqq 32t \log 16t$. □

COROLLARY 2.4. $p(G_m, t) = O(t \log t)$.

Proof. The proof is immediate from Claims 2.2 and 2.3. □

COROLLARY 2.5. *For all m, there exists a t-resilient $O(t \log t)$-agreement algorithm running on G_m for $t \leqq cn/\log n$ for some constant c.*

Proof. The proof is immediate from Claims 2.1 and 3.1. □

The next claim shows that this bound on almost-everywhere agreement is optimal for the butterfly network.

CLAIM 2.6. $p(G_m, t) = \Omega(t \log t)$.

Proof. Given t, let $k = \lceil \log t \rceil$, and choose the faulty nodes to be $\{(i, j) | i = 0, k-1, j = 0, \cdots, 2^{k-1} - 1\}$. This choice completely disconnects the set of nodes $\{(a, b) | a = 0, \cdots, k-1, b = 0, \cdots, 2^{k-1} - 1\}$, which is of size $\geqq t \log t$, from the remainder of the network. □

2.4. Almost all regular graphs have good transmission schemes. Let $H(r, n)$ be the set of regular graphs of degree r and size n. Let $h(r, n) = |H(r, n)| \geqq (n/r)^{nr/2}$[Bo]. We will show that for almost every r-regular graph G, $p(G, t) \in O(t^{1+\delta} \log t)$ for large

values of t. This is done by first showing that for all $r \geqq 5$ almost r-regular graphs G have three-phase transmission schemes for which $b(G, t) \in O(t^{1+\delta} \log t)$ for $t \in O(n^{1-\varepsilon})$, for appropriate δ, ε.

LEMMA 2.7. *For every $r \geqq 5$ there exists a constant $\alpha = \alpha(r)$, $0 < a < 1$, such that $h(r, n)(1 - O(n^{-1/4}))$ of the $h(r, n)$ r-regular graphs have the following expansion property: Every subset of vertices U such that $|U| \leqq \alpha n$ has at least $|U|(r-3)$ neighbors outside U.*

Proof. We turn the set $H(r, n)$ into a probability space by giving every graph $G \in H(r, n)$ the same probability. A proof that a random graph $G \in H(r, n)$ possesses the properties (1) and (2) with probability at least $1 - O(n^{-1})$ implies Lemma 2.7.

The study of the probability space $H(r, n)$ presents a special difficulty since no explicit representation of the set $H(r, n)$ or a direct procedure for constructing a random element in $H(r, n)$ are available. Recently, Bollobás [Bo] derived a method for studying $H(r, n)$ through a related, more simple probability space $\Phi(r, n)$. Our proof is based on this new approach.

We start with a set,

$$W(r, n) = \{(v, i) | v = 1, \cdots, n, i = 1, \cdots, r\}.$$

A configuration F is a partition of W into $rn/2$ pairs $\{(v, i), (v', i')\}$. Let $\Phi(r, n)$ be the set of all configurations of $W(r, n)$. A configuration F defines an r-degree multigraph (with possible self loops), with vertex set $V = \{1, \cdots, n\}$ in which v is connected to v' if F includes a pair $\{(v, i)\ (v', i')\}$ for some $1 \leqq i, i' \leqq r$.

Let $\Psi(r, n)$ be the set of all configurations that define a proper r-regular graph. Bollobás was able to show that $|\Psi(r, n)| \approx e^{(-r^2-1)/4} |\Phi(r, n)|$. Thus, if we turn $\Psi(r, n)$ and $\Phi(r, n)$ into probability spaces by giving all their elements equal probability, an event that holds with probability $1 - O(n^{-1})$ in $\Psi(r, n)$ also holds with probability $1 - O(n^{-1})$ in $\Phi(r, n)$. Furthermore, each graph $G \in H(r, n)$ is obtained from precisely $(r!)^n$ configurations. Thus, to prove that an event holds with probability $1 - O(n^{-1})$ in $H(r, n)$ it is enough to prove this result in the space $\Phi(r, n)$, where the analysis is substantially easier.

We first obtain a lower bound for the number of edges with at least one endpoint in a given set of vertices U. Let $\beta = \frac{3}{2}$ and $\alpha \leqq e^{-2r} r^{-4}$. Let E_1 be the event: *There is a subset of vertices U, $|U| = k \leqq \alpha n$, that are endpoints of less than $(r - \beta)k$ distinct edges.* The event E_1 implies that at least βk edges in the graph connect two vertices of U. Thus,

$$\text{Prob}\,(E_1) \leqq \sum_{k \leqq \alpha n} \binom{n}{k} \binom{rk}{\beta k} \left(\frac{k}{n}\right)^{\beta k}.$$

Since

$$\binom{n}{k} \leqq \left(\frac{ne}{k}\right)^k,$$

$$\text{Prob}\,(E_1) \leqq \sum_{k \leqq \alpha n} \left(\frac{en}{k}\right)^k (er)^{\beta k} \left(\frac{k}{n}\right)^{\beta k}.$$

For $k \leqq \log^3 n$, each of the terms in the sum is bounded by $O(n^{-1/2})$; thus, the sum of the first $\log^3 n$ terms is bounded by $O(n^{-1/3})$. For $k \geqq \log^3 n$ each term in the sum is bounded by

$$(e^{(1+\beta)} r^{\beta} \alpha^{(\beta-1)})^k \leqq n^{-2}$$

by the choice of α and β. There are less than n terms in the sum; thus, $\text{Prob}\,(E_1) = O(n^{-1/3})$.

Let E_2 be the event: *There exists a subset of vertices U, $|U| \leq \alpha n$ that has less than $|U|(r-3)$ neighbors outside U.* Conditioning on $\bar{E}_1$, the complement of the event E_1, we know that the vertices of U are endpoints of at least $(r-\beta)|U|$ edges. We bound Prob $(E_2|\bar{E}_1)$ by the probability that the other endpoints of all these edges are in a set of size $(r-3)|U|$. Thus,

$$\begin{aligned}\text{Prob}\,(E_2|\bar{E}_1) &\leq \sum_{k\leq \alpha n} \binom{n}{k}\binom{n}{(r-3)k}\left(\frac{k(r-3)}{n}\right)^{(r-\beta)k} \\ &\leq \left(\frac{ne}{k}\right)^k \left(\frac{ne}{(r-3)k}\right)^{(r-3)k}\left(\frac{(r-3)k}{n}\right)^{(r-\beta)k} \\ &\leq \left(e^{(r-2)}(r-3)^{(3-\beta)}\left(\frac{(r-3)k}{n}\right)^{(2-\beta)}\right)^k.\end{aligned}$$

Again we distinguish between two cases. If $k \leq \log^3 n$ then since $2-\beta = \frac{1}{2}$, each term in the sum is bounded by $O(n^{-1/2})$. If $k \geq \log^3 n$ then each term is bounded by $(e^r r^{3/2}(r\alpha)^{1/2})^k$, and thus, by the choice of α, is bounded by $O(n^{-2})$.

Thus, the expansion property holds with probability Prob $(\bar{E}_2) \geq$ Prob $(\bar{E}_2|\bar{E}_1)$ Prob $(\bar{E}_1) \geq 1 - O(n^{-1/3})$. □

CLAIM 2.8. *Let $G = (V, E)$ be a graph satisfying the following expansion condition: Every subset of vertices U such that $|U| \leq \alpha n$ has at least $2|U|$ neighbors outside U. Then G has the following superconcentration property: Every two subsets of vertices U, W of size $|U| = |W| \leq \alpha n$ are connected by $|U|$ paths which are vertex-disjoint except that a node in $U \cup W$, in addition to being an endpoint, might appear as an intermediate node on one other path. item.*

Proof. We use the following extension of Menger's theorem.

THEOREM [Br, p. 167]. *If a and b are two nonadjacent vertices of a simple graph G, then the maximum number of vertex-disjoint paths between a and b equals $c_G(a, b) = \min_A |A|$, where the minimization is taken over all vertex sets A that do not contain either a or b and whose removal disconnects a from b.*

Assume that our graph $G = (V, E)$ contains a pair of sets U, W that do not satisfy the property. Let $k = |U| = |W|$. Consider the graph $H = (V', E')$ obtained from G as follows: The set of vertices $V' = V + U' + W' + \{s, t\}$, $|U'| = |W'| = k$. The vertex s is connected to all u', $u' \in U'$ and the vertex t is connected to all w', $w' \in W'$. The connections between the vertices V are as in the graph G, and the vertices in $U' \cup W'$ have the same connections as the corresponding vertices in $U \cup W$. Existence of k disjoint paths connecting s to t implies the superconcentration property. The above theorem implies that if there are no k vertex-disjoint paths connecting s to t then there is a set A, $|A| < k$ that disconnects s from t.

We will prove that the existence of such a set in H violates the expansion property of G. Assume that a set A, $|A| < k$ disconnects s from t. Let $a = |U \cap A|$, $b = |V \cap A|$ and $c = |W \cap A|$, $a + b + c < k$. By the expansion property $A \cap V$ can disconnect only a set of size $\frac{1}{2}b$ from the rest of the set V. Thus, the set $V - A$ has a connected component V' of size at least $|V| - \frac{3}{2}b$. By the expansion property the set $U - A$ has at least $2(k-a)$ neighbors in V and the set $W - A$ at least $2(k-c)$ neighbors in V. Since $a + b + c < k$, $2(k-a) > \frac{3}{2}b$ and $2(k-c) > \frac{3}{2}b$; thus both $U - A$ and $W - A$ are connected to the connected component V' and there is a path connecting s to t that does not use the set A. □

CLAIM 2.9. *For all $r \geq 5$, for every graph G in $H(r, n)$, if G has the expansion and superconcentration properties, then there exists a three-phase transmission scheme for G*

subject to $p(G, t) \in O(t^{1+\delta} \log t)$ *for* $t \in O(n^{1-\varepsilon})$, *for some* $0 < \delta < \varepsilon < 1$, $\varepsilon = \varepsilon(r)$, *where* $\varepsilon \to 0$ *as* $r \to \infty$.

Proof. In this proof all logarithms are to the base $r-3$. Let $d = \lceil \log t \rceil$ and let $\delta = \log((r-1)/(r-3))$. For each vertex u in G, let $D(u)$ denote the set of vertices in G at distance $d+4$ from u. By the expansion property, and the fact that $r-3 \geqq 2$, $|D(u)| \geqq 16(r-3)^d$. We choose $\Gamma(u) = \Gamma_{\text{in}}(u) = \Gamma_{\text{out}}(u)$ to be an $s = 9(r-3)^d \leqq \alpha n$ element subset of $D(u)$. This leaves us free to use the same inbound paths as outbound paths. Specifically, we choose an arbitrary breadth-first search tree of G rooted at u, and use its branches to define paths between u and the elements of $\Gamma(u)$. Finally, by the superconcentration property there are s "almost vertex disjoint" paths between $\Gamma(v)$ and $\Gamma(w)$ for every two nodes v, w of G. For each pair of nodes we specify a particular set of such paths. This completes the specification of the transmission scheme.

It remains to analyze the damage that can be caused by the faulty processors and to show that it cannot be too large. Let u be an arbitrary correct processor and let p be a faulty processor at distance i from u. Then p can affect the paths from u to at most $(r-1)^{d+4-i}$ elements of $\Gamma(u)$. At most $r(r-1)^{i-1}$ vertices u are at distance i from p. Thus p can corrupt at most $r(r-1)^{d+3}$ elements in sets $\Gamma(u)$ for vertices u at distance i from p. Summing up for all distances $i \leqq d$ and all faulty processors we see that the faulty processors can corrupt at most $tdr(r-1)^{d+3}$ paths in total. Therefore, assuming that this damage is distributed in a worst-case fashion for the transmission scheme (optimal for an adversary),

$$b(G, t) \leqq \frac{t \log tr(r-1)^{d+3}}{9(r-3)^d/8}$$

$$= \frac{8r(r-1)^3}{9} t^{1+\delta} \log t.$$

Finally note that by the remark following Claim 2.2 and the fact that $s/8 > (r-3)^d \geqq t$, the same bound applies for $p(G, t)$. Note also that this bound limits the results to $t \in O(n^{1-\varepsilon})$ for an appropriate $\delta < \varepsilon < 1$. □

COROLLARY 2.10. *For every* $r \geqq 5$ *almost every r-regular graph has a t-resilient* $O(t^{1+\delta} \log t)$*-agreement algorithm, for* $t \in O(n^{1-\varepsilon})$, *for some* $0 < \delta < \varepsilon < 1$. □

2.5. Results for faults of restricted severity. In this section we briefly mention our results for the failstop, omission, and authenticated Byzantine models. As with the unrestricted Byzantine failures (without authentication) our approach is to devise an appropriate transmission scheme. However, in the failure models considered here two correct processors can communicate provided they are connected by even one uncorrupted path. Thus the poor processors will be only those completely disconnected from the major portion of the graph. Now, if a graph has good expansion properties, then it is impossible for t processors to disconnect more than $O(t)$ processors. Combining this with the fact that almost all regular graphs of degree at least 5 enjoy such properties (Lemma 2.7), we obtain the following result.

THEOREM 2.11. *Assuming an authentication mechanism or nonmalicious faults (failstop, omission), for all* $r \geqq 5$ *there exists a constant* $c = c(r)$ *such that for all* $t \leqq cn$,

(1) *almost all r-regular graphs have transmission schemes in which there are at most* $O(t)$ *poor processors, and*

(2) *almost all r-regular graphs admit a t-resilient algorithm for* $O(t)$ *agreement.*

3. Byzantine agreement on compressor graphs. Until this point we have considered only techniques based on simulations of complete-network algorithms on our bounded

degree networks. A different approach to reaching agreement in a bounded degree network can be based on distributed protocols of a local nature. In this section we consider such an approach and analyze its behavior on a compressor graph.

A graph G is a *θ-compressor* for constant $\theta < 1$, if for every subset of vertices U of size $|U| \leq \theta n$, the set $\{v | v$ has at least half of its neighbors in $U\}$ has cardinality at most $|U|/2$. An explicit construction of compressors of a (very high) fixed degree (about 8^{17}) has been shown by Pippenger [P]. Using a recent result of Lubotzky, Phillips, and Sarnak [LPS] the degree of explicitly constructed compressors can be reduced to about 64.

Our interest in compressors follows from our ability to use the compression property to "sharpen" almost-everywhere agreement to achieve $O(t)$ agreement (which is asymptotically optimal, as t faults can always completely disconnect $\Omega(t)$ correct processors). Ultimately, our approach will be to use one of the simulation techniques of the previous section to obtain $p(G, t)$ agreement and then to sharpen this to $O(t)$ agreement by using the compressor properties of the graph.

We consider the following scheme for agreement, called the *compression procedure.* This procedure is based on simple rounds of the following form. In every round, every correct processor

(1) sends its value to all its neighbors,

(2) receives the values of all its neighbors, and

(3) chooses as its new value the value held by a majority of its neighbors.

The procedure terminates after some fixed number of rounds.

Let $G = (V, E)$ be a θ-compressor of size n, and let T be the set of faulty processors in the network, $|T| \leq t \leq \theta n/2$.

LEMMA 3.1. *If there are $(1-\theta)n$ correct processors which share the same initial value x, then after applying the compression procedure for $\log n$ rounds, at most $t+1$ correct processors will have a value different from x.*

Proof. Let V_k denote the set of correct processors whose value differs from v after k rounds of the majority procedure. Let $A_k = ((2^k - 1)t + |V_0|)/2^k$, for $k \geq 0$. We begin by bounding A_k in terms of n and t.

CLAIM 3.1.1. *For every $k \geq 0$, $A_k + t \leq \theta n$.*

Proof. There are two cases. If $|V_0| \leq t$ then for every $k \geq 0$, $A_k \leq ((2^k - 1)t + t)/2^k = t$ and we are done by the assumption on t. If $|V_0| > t$ we prove the claim by induction on k. For $k = 0$, $A_k = |V_0|$, and $A_k + t = |V_0| + t \leq \theta n$ by the assumption on the initial state of the network. Assume the claim inductively for k. Since $A_{k+1} = A_k + (t - |V_0|)/2^{k+1} < A_k$ we are done. □

We next bound V_k in terms of A_k.

CLAIM 3.1.2. *For every $k \geq 0$, $|V_k| \leq A_k$.*

Proof. We prove the claim by induction on k. For $k = 0$ the claim is trivial. Assume the claim for k inductively, and consider the case of $k+1$. The set V_{k+1} will contain all correct processors whose majority of neighbors resides in $T \cup V_k$. By the inductive hypothesis this set has cardinality at most $t + A_k$, which, by Claim 3.1.1, is at most θn. We may therefore apply the compression property to obtain $|V_{k+1}| \leq (t + A_k)/2 = A_{k+1}$.

Taking $k = \log n$ we obtain $|V_k| \leq A_k \leq t + 1$. This completes the proof of Claim 3.1.1. □

LEMMA 3.2. *For every $r \geq 5$ there exists a constant $0 < \theta < 1$, such that $h(r, n)(1 - O(n^{-1}))$ of the $h(r, n)$ r-regular graphs have the compression property for all subsets U of size $|U| \leq \theta n$.*

Proof. We use the same setting and notation as in the proof of Lemma 2.7. Define the event E_3: *There exists a set U, $|U| \leq \theta n$ and a set W, $|W| \leq |U|/2$ such that half of*

the edges incident on W have one endpoint in W and the other in the set U. Clearly if a graph has a set U that violates the compression property it is included in the event E_3. To estimate the probability of the event E_3 we condition first on the complement of the event E_1 (defined in the proof of Lemma 2.7). Under the condition $\bar{E}_1$, the vertices of the set W are incident to at least $(r-\beta)|W|$ edges, and to satisfy the condition E_3 at least half of these edges have their other endpoint in the set U:

$$\begin{aligned}\text{Prob}(E_2|\bar{E}_1) &\leqq \sum_{k\leqq\theta_n}\binom{n}{k}\binom{n}{k/2}\binom{(r-\beta)k}{((r-\beta)/2)k}\left(\frac{rk}{rn}\right)^{((r-\beta)/2)k}\\ &\leqq \left(\frac{ne}{k}\right)^{k}\left(\frac{2ne}{k}\right)^{k/2}\left(\frac{(r-\beta)k}{((r-\beta)/2)k}\right)^{(r-\beta)k}\left(\frac{k}{n}\right)^{((r-\beta)/2)k}\\ &\leqq \left(e^{3k/2}2^{(r-\beta+1/2)}\left(\frac{k}{n}\right)^{(r-\beta-3/2)}\right)^{k}\end{aligned}$$

For $k\leqq\log^3 n$ each term in the sum is bounded by $O(n^{-1})$. For $k\geqq\log^3 n$ each term is bounded by $(e^{r+1/2}\theta^{r-3/2})^k\leqq n^{-2}$, since $\beta=\frac{3}{2}$ and e^{-2}. Thus, $\text{Prob}(\bar{E}_3)\geqq \text{Prob}(\bar{E}_3|\bar{E}_1)\,\text{Prob}(\bar{E}_1)\geqq 1-O(n^{-1})$. □

Combining this with the transmission schemes described in § 2.4, we prove the following.

COROLLARY 3.3. *For every $r\geqq 5$, $h(r,n)(1-O(n^{-1}))$ of the $h(r,n)$ r-regular graphs admit $O(t)$ agreement, where $t\in O(n^{1-\varepsilon})$ for some $0<\varepsilon<1$.*

Proof. By Lemmas 2.7 and 3.2 almost every r-regular graph enjoys the expansion, compression, and superconcentration properties. Let G be any such graph. Using the three-phase transmission scheme of § 2.4 to simulate any standard Byzantine agreement designed for the complete network, we achieve $O(t^{1+\delta}\log t)$ agreement, so at the end of the simulation at least $1-\theta$ of the correct processors agree on a value. Since this satisfies the conditions of Lemma 3.1 we may apply that lemma to obtain the desired $O(t)$-agreement. □

COROLLARY 3.4. *There exists a family of degree 9 networks that admit t-resilient $O(t)$ agreement for $t\leqq\alpha n/\log n$, for some constant α.*

Proof. Since a butterfly has degree 4 and a compressor has degree 5, there exists a graph of degree 9 which is both a butterfly and a compressor. We first use the result of Corollary 2.5 to get an $O(t\log t)$ agreement on the butterfly graph, then we use the compressor to sharpen this agreement. □

4. Almost-everywhere agreement on networks of unbounded degree. On a complete network, t-resilient agreement protocols exist for all $t<n/3$. In contrast, our previous algorithms can tolerate only $O(n/\log n)$ failures. The situation for higher values of t on bounded degree networks remains open. In this section we present an approach for reaching agreement on networks of unbounded but small degree (which is nevertheless much lower than n or t).

THEOREM 4.1. *For every $\varepsilon>0$ there exist a constant $c=c(\varepsilon)$, graphs G of degree $O(n^{\varepsilon})$ and t-resilient $O(t)$-agreement algorithms for $t\leqq cn$.*

Proof. For simplicity, we discuss only the case $\varepsilon=\frac{1}{2}$. Let $G=(V,E)$ be defined as follows. V contains $n=m^2$ nodes, partitioned into m pairwise disjoint committees A_i, $1\leqq i\leqq m$, each of size m. Each committee forms a clique. In addition, every two committees A_i and A_j are connected by m edges which form a matching between them. On top of that, we optionally add edges so as to make the graph into a θ-compressor. As discussed earlier, this property can be achieved by a graph of bounded degree. Thus the degree of any node in the resulting graph will be $O(n^{1/2})$.

Let P be any t-resilient protocol for the traditional Byzantine agreement problem on a complete network [DFFLS], [PSL], [LSP], guaranteeing agreement whenever the number of faulty processors is smaller than $n/3$.

The algorithm we give consists of two main stages plus an additional optional stage which uses the compressor edges. In the first stage, each committee privately runs P (appropriately scaled) among its own members. We say a committee is *good* if fewer than $\frac{1}{4}$ of its members are faulty. Clearly the good committees will reach agreement among themselves. In the next stage, each committee acts as a single processor and the whole graph is viewed as a clique of m (composite) processors which simulate an execution of P. The final optional stage is a compression step, as described in § 3.

The communication between committees is performed by a special protocol COM which guarantees that processors within a good committee behave uniformly. According to this protocol a committee A sends a message x to B as follows. Every processor in A sends x to its neighbor in B. Every node in B now executes the following two steps:

(1) Exchanges with all other processors in B the value received from A;

(2) Adopts as its updated value of x that message received from the majority of the nodes in B (ties are broken arbitrarily).

Finally, the nodes of B run P on the adopted values for x. The value agreed upon by the end of the run is taken to be the message sent by A.

We now analyze the behavior of this algorithm. Call a committee A_i *good* if $t_i < m/4$, where t_i is the number of actual faults in A_i, and *bad* otherwise.

CLAIM 4.2. *In every communication step (i.e., every execution of* COM*), if the committee A_i receiving a message is good then all the good processors in it will agree on the same received value.*

Proof. This follows from the properties of P and the fact that $t_i < m/3$. □

COROLLARY 4.3. *In every round of the second stage of the main algorithm, the elements of a good committee will have the same view and will send out the same messages.* □

CLAIM 4.4. *In every communication step* COM*, if the receiving committee A_i and the sending committee A_j are good, then all the good processors in A_j will agree on the value that was sent by (the good processors of) A_i.*

Proof. Since all the good processors of A_i send the same message X, and $t_i, t_j < m/4$, the number of good processes in A_j receiving X is at more than $m/2$. Therefore after the majority step of COM, all the correct processes in A_j will have the same value X. By the unanimity property of the algorithm P, all the correct processors in A_j will eventually agree on X. □

CLAIM 4.5. *If $t < n/12$ then in the end of the basic main algorithm (without the final compression step), agreement will be reached between most of the good processors; at most $3t$ good processors will reach a wrong decision.*

Proof. Denote the number of bad committees by b. Clearly $b \leqq 4t/m$ (otherwise, since every bad committee has at least $m/4$ faults, there are more than t faults overall). Hence $b < (4/m)(n/12) = m/3$. Therefore the main algorithm will guarantee that all the good committees will reach agreement, where in every good committee all good processors will have the right value. The "confused" good processors are at most those in the bad committees, whose number is bounded by $b(m - m/4) \leqq (3m/4)(4t/m) = 3t$. □

CLAIM 4.6. *If $t < \theta n/4$ then in the end of the algorithm (with the final compression step), agreement will be reached between most of the good processors; at most $t+1$ good processors will reach a wrong decision.*

Proof. Upon termination of the basic algorithm, the number l of good processors holding a wrong value is bounded by $3t$, as shown in the previous claim. Thus $t < \theta n/2$, and $l + t \leqq 4t < \theta n$, and by Claim 2.3, after the compression procedure at most $t+1$ good processors will have a wrong value. □

This algorithm can be naturally extended for graphs of degree $O(n^{1/k})$ for $k = 3, 4, \cdots$, by dividing the graph into committees of committees of committees, etc., with an appropriate definition of a good committee on every level. We omit the details.

5. Combinatorial characterization of fault-tolerant networks. There is a necessary and sufficient condition for a system to be t-resilient in terms of the combinatorial properties of its communication graph. In this section we derive a combinatorial characterization of graphs that admit $p(t)$ agreement.

For any agreement protocol P let $P(T)$ be any maximal set of correct processors that always reach agreement under the protocol P, independent of the behavior of the processors in T (thought of as faulty).

THEOREM 5.1. *Let G be a communication graph, let $\{T_i\}_{i=1}^{k}$ be the family of all possible sets of faulty processors in G, and let $\{A(T_i)\}_{i=1}^{k}$ be a family of sets of processors in G. There exists a protocol P subject to $P(T_i) = A(T_i)$ for $i = 1, \cdots, k$, if and only if for every pair of processors $u, v \in A(T_i) \cap A(T_j)$, the set $T_i \cup T_j$ does not disconnect u from v in G.*

Comment. Not that if for all sets T of at most t processors $A(T)$ is the complement of T, then our characterization yields the $2t+1$ connectivity condition for the traditional Byzantine agreement problem.

Sketch of proof. To prove necessity, we show that if there exist sets T_i and T_j which jointly (but not individually) can disconnect correct processors u and v, then there exist two scenarios, indistinguishable to v, such that in one scenario T_i is faulty and u decides on a value a, while in the second scenario T_j is faulty and u decides on a value b.

We prove sufficiency by constructing an algorithm with the desired properties. We briefly describe a few points of our construction.

A processor u transmits a message to v by sending it along all simple paths from u to v. As the message passes from site to site, each processor appends the name of the processor from which the message was received. Thus, a message that passes through faulty processors contains the name of at least one such processor (the last one). The processor v searches for a set T_i such that all the messages not passing through this set are consistent and both u and v are in $A(T_i)$. Let T be the set of faulty processors in a particular execution of our algorithm. If u and v are in $A(T)$ then v will try this set and extract the correct value. Crucial to our algorithm is that v will never extract an incorrect value. This is because by assumption, for all other relevant sets T_j, $T \cup T_j$ will not disconnect u from v. Thus, v receives the message via at least one fault-free path. Therefore, the faulty processors can at most create an inconsistent set of values, from which v extracts nothing. □

6. Conclusions and open problems. We propose a new paradigm for fault-tolerant computing, in which correctness conditions are relaxed by accepting the loss of a small number of correct processors.

This paper concentrates on demonstrating the feasibility of this new approach and its advantages. In particular, we show that a simple distributed algorithm can achieve agreement among almost all the correct processors in circumstances in which Byzantine agreement is not achievable.

Many problems remain open for further investigation. We mention the following:

(1) What is the minimum number of steps and messages required to achieve X agreement? Can the efficiency of the algorithms presented in this paper be significantly improved?

(2) The algorithms presented in this work assume that all the processors know the topology of the communication network and the communication schemes used by all other processors. Is this requirement essential for achieving X agreement?

(3) Can X agreement be achieved on a network of bounded degree in the presence of more than $n/\log n$ (malicious) faults? Note that in this case most of the communication paths between most pairs of processors include at least one faulty processor.

Acknowledgment. We wish to thank M. Ajtai for helpful discussions.

REFERENCES

[Br] C. BERGE, *Graphs and Hypergraphs*, second edition, North-Holland, Amsterdam, 1979.

[Bo] B. BOLLOBÁS, *Random graphs*, in Combinatorics, London Mathematical Society Lecture Notes 52, Cambridge University Press, Cambridge, 1981, pp. 80–102.

[D] D. DOLEV, *The Byzantine generals strike again*, J. Algorithms, 3 (1982), pp. 14–30.

[DFFLS] D. DOLEV, M. J. FISCHER, R. FOWLER, N. A. LYNCH, AND R. STRONG, *An efficient algorithm for Byzantine agreement without authentication*, Inform. and Control, 52 (1982), pp. 256–274.

[F] M. J. FISCHER, *The consensus problem in unreliable distributed systems*, Tech. Report 273, Dept. of Computer Science, Yale University, New Haven, CT, 1983.

[GG] O. GABBER AND Z. GALIL, *Explicit construction of linear-sized superconcentrators*, J. Comput. System Sci., 22 (1981), pp. 407–420.

[H] V. HADZILACOS, *Issues of fault tolerance in concurrent computations*, Ph.D. thesis, Harvard University, Cambridge, MA, 1984.

[LPS] A. LUBOTZKY, R. PHILLIPS, AND P. SARNAK, *Ramanujan conjecture and explicit constructions of expanders*, in Proc. 18th Annual ACM Symposium on Theory of Computing, 1986, pp. 240–246.

[LSP] L. LAMPORT, R. SHOSTAK, AND M. PEASE, *The Byzantine generals problem*, ACM Trans. Prog. Lang. Syst., 4 (1982), pp. 382–401.

[PSL] M. PEASE, R. SHOSTAK, AND L. LAMPORT, *Reaching agreement in the presence of faults*, J. ACM, 27 (1980), pp. 228–234.

[P] N. PIPPENGER, *On networks of noisy gates*, in Proc. 26th Annual ACM Symposium on Foundations of Computer Science, 1985, pp. 31–38.

[U] J. D. ULLMAN, *Computational Aspects of* VLSI, Computer Science Press, Potomac, MD, 1984.