The Effect of Faults on Network Expansion

Amitabha Bagchi * Ankur Bhargava† Amitabh Chaudhary‡ David Eppstein§

Christian Scheideler†

October 1, 2004

Abstract

In this paper we study the problem of how resilient networks are to node faults. Specifically, we investigate the question of how many faults a network can sustain so that it still contains a large (i.e. linear-sized) connected component that still has approximately the same expansion as the original fault-free network. For this we apply a pruning technique which culls away parts of the faulty network which have poor expansion. This technique can be applied to both adversarial faults and to random faults. For adversarial faults we prove that for every network with expansion α, a large connected component with basically the same expansion as the original network exists for up to a constant times $\alpha \cdot n$ faults. This result is tight in the sense that every graph G of size n and uniform expansion $\alpha(\cdot)$, i.e. G has an expansion of $\Theta(\alpha(n))$ and every subgraph G' of size m of G has an expansion of $O(\alpha(m))$, can be broken into sublinear components with $\omega(\alpha(n) \cdot n)$ faults.

For random faults we observe that the situation is significantly different. In this case the expansion of a graph only gives a very weak bound on its resilience to random faults. Specifically, there are networks of uniform expansion $O(1/\sqrt{n})$ that are resilient against a constant fault probability but there are also networks of uniform expansion $\Omega(1/\log n)$ that are not resilient against a $O(1/\log n)$ fault probability. Thus, a different parameter is needed. For this we introduce the *span* of a graph which allows us to determine the maximum fault probability in a much better way than the expansion can. We use the span to show the first known results for the effect of random faults on the expansion of d-dimensional meshes.

1 Introduction

Communication in faulty networks is a classical field in network theory. In practice, one cannot expect nodes or communication links to work without complications. Software or hardware faults (or phenomena outside the control of a network operator such as caterpillars) may cause nodes or links to go down. To be able to adapt to faults without a serious degradation of the service, networks and routing protocols have to be set up so that they are fault-tolerant. Fault-tolerant routing has recently attained renewed interest due to the tremendous rise in popularity of mobile ad-hoc networks and peer-to-peer networks. In these networks, faults are actually not an exception but a frequently occurring event: in mobile ad-hoc networks, users may run out of battery power or may move out of reach of others, and in peer-to-peer networks, users may leave without notice.

Central questions in the theoretical area of faulty networks have been:

- How many faults can a network sustain so that the size of its largest connected component is still a constant fraction of the original size?
- How many faults can a network sustain so that it can still emulate its ideal counterpart with constant slowdown?

The first question has been heavily studied in the graph theory community, and the second question has been investigated mostly by the parallel computing community to find out up to which point a faulty parallel computer can

*Dept. of Computer and Information Science, Polytechnic University, 6 Metrotech Center, Brooklyn, NY 11201. bagchi@cis.poly.edu.

†Dept. of Computer Science, Johns Hopkins University, Baltimore, MD 21218. {ankur,scheideler}@cs.jhu.edu

‡Dept. of Computer Science and Engineering, Notre Dame University, Notre Dame, IN 46556. achaudha@cse.nd.edu

§Dept of Computer Science, University of California, Irvine, 92697. eppstein@ics.uci.edu

still emulate an ideal parallel computer with the same topology with constant slowdown. We refer the reader to [27] for a survey of results in these areas.

In this paper we navigate a path between these two extremes. The fact that the network is still connected gives little solace if the connection is a tight bottleneck. On the other end, we might not *need* to emulate a fault-free network, all we might need is that the network's routing properties do not degrade excessively. Specifically we study the effect of faults on the expansion of a network, showing bounds on the number of adversarial faults and the fault probability that a network can take and still retain a large component with good enough expansion.

But before we proceed to our results we discuss previous work related to connectivity and emulation in the face of faults.

1.1 Large connected components in faulty networks

We start with an overview of previous results for random faults and afterwards consider adversarial faults.

Given a graph G and a probability value p, let $G^{(p)}$ be the random graph obtained from G by keeping each edge of G alive with probability p (i.e. p is the *survival probability*). Given a graph G, let $\gamma(G) \in [0,1]$ be the fraction of the nodes of G contained in a largest connected component.

Let $\mathcal{G} = \{G_n \mid n \in \mathbb{N}\}$ be any family of graphs with parameter n. Let p^* be the *critical probability* for the existence of a linear-sized connected component. I.e. for every constant $\epsilon > 0$ it holds:

1. For every $p > (1+\epsilon)p^*$ there exists a constant $c > 0$ with $\lim_{n\to\infty} \Pr[\gamma(G_n^{(p)}) > c] = 1$.
2. For all constants $c > 0$ and for all $p < (1-\epsilon)p^*$ it holds that $\lim_{n\to\infty} \Pr[\gamma(G_n^{(p)}) > c] = 0$.

Of course, it is not obvious whether critical probabilities exist. However, the results by Erdős and Rényi [11] and its subsequent improvements (e.g. [6, 22]) imply that for the complete graph on n nodes, $p^* = 1/(n-1)$, and that for a random graph with $d \cdot n/2$ edges, $p^* = 1/d$. For the 2-dimensional $n \times n$-mesh, Kesten showed that $p^* = 1/2$ [17]. Ajtai, Komlós and Szemerédi proved that for the hypercube of dimension n, $p^* = 1/n$ [1]. For the n-dimensional butterfly network, Karlin, Nelson and Tamaki showed that $0.337 < p^* < 0.436$ [16]. Leighton and Maggs [18] showed that there is an indirect constant-degree network connecting n inputs with n outputs via $\log n$ levels of n nodes each, called multibutterfly, that has the following property: Up to a constant fault probability it is still possible to find $O(\log n)$ length paths from a constant fraction of the inputs to a constant fraction of the outputs. Subsequently Cole, Maggs and Sitaraman [7] extended this result for the butterfly.

Adversarial fault models have also been investigated. Leighton and Maggs [18] also showed that no matter how an adversary chooses f nodes to fail, there will be a connected component left in the multibutterfly with at least $n - O(f)$ inputs and at least $n - O(f)$ outputs. (In fact, one can even still route packets between the inputs and outputs in this component in almost the same amount of time steps as in the ideal case.) Subsequently Leighton, Maggs and Sitaraman [20] extended this result for the butterfly.

Upfal [28], following up on work by Dwork et. al. [10] and Alon and Chung [2], showed that there is also a direct constant-degree network on n nodes, a so-called expander, that has the property: no matter how an adversary chooses f nodes to fail, there will be a connected component left in it with at least $n - O(f)$ nodes. Both results are optimal up to constants. Upfal uses a pruning technique to achieve his bound which is similar in spirit to the one we use. Apart from the fact that Upfal gives a polynomial-time algorithm for pruning while we do not, the important difference worth noting is that Upfal's pruning does not guarantee a large component of good expansion. In fact, to the best of our knowledge there is no known constant approximation algorithm to determine the expansion of a graph of unknown topology.

1.2 Simulation of fault-free networks by faulty networks

Next we look at the problem of simulating fault-free networks by faulty networks. Consider the situation that there can be up to f worst-case node faults in the system at any time. One way to find out whether the largest remaining component still allows efficient communication is to check whether it is possible to embed into the largest connected component of a faulty network a fault-free network of the same size and kind. An *embedding* of a graph G into a graph H maps the nodes of G to non-faulty nodes of H and the edges of G to non-faulty paths in H. An embedding is called *static* if the mapping of the nodes and edges is fixed. Both static and dynamic embeddings have been used. A good embedding is one with minimum load, congestion, and dilation, where the *load* of an embedding is the maximum

number of nodes of G that are mapped to any single node of H, the *congestion* of an embedding is the maximum number of paths that pass through any edge e of H, and the *dilation* of an embedding is the length of the longest path. The load, congestion, and dilation of the embedding determine the time required to emulate each step of G on H. In fact, Leighton, Maggs, and Rao have shown [19] that if there is an embedding of G into H with load ℓ, congestion c, and dilation d, then H can emulate any communication step (and also computation step) on G with slowdown $O(\ell + c + d)$.

When demanding a constant slowdown, only a few results are known so far. In the case of worst-case faults, it was shown by Leighton, Maggs and Sitaraman (using dynamic embedding strategies) that an n-input butterfly with $n^{1-\epsilon}$ worst-case faults (for any constant ϵ) can still emulate a fault-free butterfly of the same size with only constant slowdown [20]. Furthermore, Cole, Maggs and Sitaraman showed that an $n \times n$ mesh can sustain up to $n^{1-\epsilon}$ worst-case faults and still emulate a fault-free mesh of the same size with (amortized) constant slowdown [9]. It seems that also the n-node hypercube can achieve a constant slowdown for $n^{1-\epsilon}$ worst-case faults, but so far only partial answers have been obtained [20].

Random faults have also been studied. For example, Håstad, Leighton and Newman [13] showed that if each edge of the hypercube fails independently with any constant probability $p < 1$, then the functioning parts of the hypercube can be reconfigured to simulate the original hypercube with constant slowdown. Leighton, Maggs and Sitaraman [20] showed that a butterfly network whose nodes fail with some constant probability p can still emulate a fault-free butterfly of the same size with slowdown $2^{O(\log^* n)}$. Interestingly, in the conference version of [9], Cole, Maggs and Sitaraman claim that an $n \times n$ mesh in which each node is faulty independently with a constant fault probability is able to emulate a fault-free mesh with a constant slowdown [8]. The proof of this claim, which is stronger than the theorem we prove about the $n \times n$ mesh in this paper, is omitted in [8] and has not appeared elsewhere to the best of our knowledge.

For a list of further references concerning embeddings of fault-free into faulty networks see the paper by Leighton, Maggs and Sitaraman [20].

1.3 Our approach

The two common approaches – connectivity and emulation of fault-free by faulty networks – are too extreme for many practical applications. Knowing how long a network is still connected may not be very useful, because in extreme cases (just a single line connects one half to the other) the speed of communication may be reduced to a crawl, making it useless for applications that need a fast interaction or a large bandwidth such as interactive gaming or video conferences. On the other hand, emulating a fault free network on a faulty network is like using a giant hammer to crack a lesser nut, so to speak. Emulation may not be needed when all we want is reduced congestion or good expansion.

Applications in ad-hoc networks or peer-to-peer systems usually do not care about how a network is connected, concerning themselves instead with whether it still provides sufficient bandwidth and ensures sufficiently small delays. In this scenario a more relevant question is:

> *How many faults can a network sustain so that it still contains a network of at least a constant fraction of its original size that still has approximately the same expansion?*

Knowing an answer to this question would have many useful consequences for distributed data management, routing, and distributed computing. Research on load balancing has shown that if the expansion basically stays the same, the ability of a network to balance single-commodity or multi-commodity load basically stays the same, and this ability can be exploited through simple local algorithms [12, 3]. Also, the ability of a network to route information is preserved because it is closely related to its expansion [26]. Furthermore, as long as the original network still has a large connected component of almost the same expansion, one can still achieve almost everywhere agreement which is an important prerequisite for fundamental primitives such as atomic broadcast, Byzantine agreement, and clock synchronization [10, 28, 5].

Many different fault models have been studied in the literature: faults may be permanent or transient, nodes and/or edges may break down, and faults may happen at random or may be caused by an adversary or attacker. The former faults are called *random faults*, and the latter faults are called *adversarial faults*. We will concentrate on situations in which there are static node faults. Nodes either break down randomly or due to some adversary. A central parameter we will use in our investigations is the expansion.

Given a graph $G = (V, E)$ and a subset $U \subseteq V$, the *expansion* of U is defined as

$$\alpha(U) = \frac{|\Gamma(U)|}{|U|}$$

where $\Gamma(U)$ is the set of nodes in $V \setminus U$ that have an edge from U and $|S|$ denotes the size of set S. The *expansion* of G is defined as $\alpha(G) = \min_{U,|U|\leq|V|/2} \alpha(U)$. If G is clear from the context, we will also just write α.

We also use the *uniform expansion* of a graph. Suppose that we have an infinite family of graphs $\mathcal{G}$. Let β be a function such that $\beta(|G|) \leq \alpha(G)$ for all graphs $G \in \mathcal{G}$, where $|G|$ denotes the number of nodes in G. Then we say that $\mathcal{G}$ has a uniform expansion of β if for all subgraphs H of all graphs G in $\mathcal{G}$ it holds that $\alpha(H) = O(\beta(|H|))$.

Notice that $\alpha(H) = \min_{U,|U|\leq|V(H)|/2} \alpha(U)$, i.e. it should not be confused with $\alpha(V(H))$ in G. The uniform expansion definition implies that the expansion of a subgraph H of G cannot be asymptotically better than the expansion of G (which also implies that $\beta(|G|) = \Theta(\alpha(G))$ for all $G \in \mathcal{G}$).

Despite its seemingly complex definition, the uniform expansion property is a quite natural property. In fact, all standard graphs such as trees, d-dimensional meshes, the hypercube, the butterfly, and the deBruijn graph fulfill this uniform expansion property.

Uniform expansion is well illustrated by the example of the mesh. If we consider the $n \times n$-mesh (i.e. the 2-dimensional mesh of side length n) we can see it has uniform expansion and the function $\beta(l) = \sqrt{l}$. This is demonstrated by observing that any subgraph of size m in that mesh can have a bisection width of at most $\sqrt{m}$ and therefore an expansion of at most $O(1/\sqrt{m})$.

1.4 Our main results

Adversarial faults

We give general upper and lower bounds for the number of node faults a graph can sustain so that it still has a large component with basically the same expansion, where the bounds are tight up to a constant factor. More specifically, we show that the number of adversarial node faults a graph with expansion α and n nodes can sustain, with only a constant factor decrease in its expansion, is a constant times $\alpha \cdot n$. For (families of) graphs G of size n and uniform expansion β, this result is best possible up to a constant factor in a sense that $\omega(\beta(n) \cdot n)$ faults can break G into sublinear components.

Random faults

We also study random faults. Our main contribution here is to suggest a new parameter for their study, which may be of independent interest.

Consider a graph $G = (V, E)$. Let $U \subseteq V$ be any subset of nodes. U is defined to be *compact* if and only if U and $V \setminus U$ are connected in G. Let $\mathcal{U}$ be the set of all compact sets of G. Let $P(U)$ be the smallest tree in G which connects every node in $\Gamma(U)$ (i.e. it essentially spans the boundary of U). Note that the set of nodes in $P(U)$ need not be from U alone or from $V \setminus U$ alone. Then the *span* of a graph is defined as:

$$\sigma = \max_{U \in \mathcal{U}} \left\{ \frac{|P(U)|}{|\Gamma(U)|} \right\} \tag{1}$$

The span helps us characterize the resilience of the expansion to random faults. We show that a graph with maximum degree δ, span σ, and expansion α (fulfilling certain, weak conditions) can tolerate a fault probability of up to a constant times $1/\delta^{O(\sigma)}$ and still retain an expansion within a factor of δ of its original expansion.

We also show that the d-dimensional meshes have a constant span. The proof of this theorem is of independent value as it establishes an interesting property of the d-dimensional mesh: The boundary of any set of connected vertices in the d-dimensional mesh, whose complement is also connected, can be spanned by a tree of size at most twice the size of the boundary.

1.5 Outline of the paper

The rest of the paper is organized as follows: In Section 2 we consider adversarial faults, and in Section 3 we consider random faults. The paper ends in Section 4 with a discussion of how our results are related to previous research and some open problems.

2 Adversarial faults

In this section we prove the existence of a large connected component with good expansion in a graph with faulty nodes. We assume that a malicious adversary decides which nodes are faulty. More formally, we are given a graph $G = (V, E)$ with n nodes and expansion α. An adversary gives us a faulty version of this graph, called $G_f = (V, E_f)$, with f faulty nodes removed, where an edge $\{u, v\} \in E$ remains in E_f if and only if both u and v are non-faulty. We will show that there exists a subgraph of G_f called H which has $\Theta(n)$ nodes and has an expansion of $\Theta(\alpha)$ provided that the adversary is given no more than $O(\alpha \cdot n)$ faults.

2.1 Upper bound on adversarial faults

We cannot argue that the expansion of G_f is no more than a constant factor less than α for the simple reason that the adversary can disconnect parts of the graph, causing the expansion of G_f to be 0. The subgraph H is constructed using an algorithm called *Prune* described in Figure 1. Note that the running time of *Prune* is not necessarily polynomial, nor are we claiming it is. *Prune* simply helps us prove an existential result. We discuss the algorithmic aspects of *Prune* in greater detail in Section 2.3

Before we get to the algorithm we need to introduce some notation. Given a graph G, we define $V(G)$ to be its node set and $\Gamma_G(S)$ to be the set of nodes in the neighbourhood of a node set S in G. The algorithm generates a sequence of graphs G_0 to G_m, where the final graph G_m is the graph H we are looking for.

Algorithm *Prune*(ϵ)

1: $G_0 \leftarrow G_f$; $i \leftarrow 0$
2: **while** $\exists S_i \subseteq V(G_i)$ such that $|\Gamma_{G_i}(S_i)| \leq \alpha \cdot \epsilon \cdot |S_i|$ and $|S_i| \leq |V(G_i)|/2$
3: $\quad G_{i+1} \leftarrow G_i \setminus S_i$
4: $\quad i \leftarrow i + 1$
5: **end while**
6: $H \leftarrow G_i$; $m \leftarrow i$

Figure 1: The pruning algorithm

Theorem 2.1 *Given a graph G with n nodes, expansion α and f faulty nodes chosen by an adversary, it holds for any constant k such that $k \geq 2$ and $f \leq \frac{\alpha n}{4k}$ that* Prune$(1 - \frac{1}{k})$ *returns a subgraph H of size at least $n - \frac{f \cdot k}{\alpha}$ with expansion at least $(1 - \frac{1}{k}) \cdot \alpha$.*

Proof. Denote $G_f \setminus H$ as $\mathcal{S}$. $\mathcal{S}$ is thus the union of all the regions culled by *Prune*. To prove the result we will first show that the size of $\mathcal{S}$ is bounded by $\frac{k \cdot f}{\alpha}$ (which is at most $n/4$ according to the theorem). To show this we will use the fact that the number of faults required to cull a region is proportional to the size of the region. To demonstrate that we need the following lemma.

Lemma 2.2

$$\left| \Gamma(\bigcup_{0 \leq i \leq j} S_i) \right| \leq \sum_{0 \leq i \leq j} |\Gamma_{G_i}(S_i)| \leq \alpha \cdot (1 - \frac{1}{k}) \cdot \left| \bigcup_{0 \leq i \leq j} S_i \right|.$$

Proof. Consider the first inequality. Obviously, any node v that lies in the neighborhood of $\bigcup_i S_i$ in G must lie in the neighborhood of some S_i in G. Thus, because v is outside of $\bigcup_i S_i$ and therefore belongs to H, there must be an S_i with $v \in \Gamma_{G_i}(S_i)$. Therefore, $\Gamma(\bigcup_i S_i) \subseteq \bigcup_i \Gamma_{G_i}(S_i)$. Hence the first inequality. Each set S_i that is culled by *Prune*$(1 - \frac{1}{k})$ has the property that $|\Gamma_{G_i}(S_i)| \leq \alpha \cdot (1 - \frac{1}{k}) \cdot |S_i|$. Since the sets S_i are disjoint, $\sum_i |S_i| = |\bigcup_i S_i|$. Hence the second inequality. □

We will show that $\mathcal{S} \leq \frac{k \cdot f}{\alpha}$ by contradiction. Let, if possible, $\mathcal{S} > \frac{k \cdot f}{\alpha}$. Since at every iteration of the algorithm we pick an S_i which is the smaller side of the cut we have found, each S_i is at most $n/2$ in size. Now, since $\frac{k \cdot f}{\alpha} \leq \frac{n}{4}$,

there is a j such that either $\frac{k \cdot f}{\alpha} < \left|\bigcup_{0 \leq i \leq j} S_i\right| \leq n/2$ or $\frac{k \cdot f}{\alpha} < |S_j| \leq n/2$. So we can always choose an $\mathcal{S}' \subseteq \mathcal{S}$ such that $\frac{k \cdot f}{\alpha} < |\mathcal{S}'| \leq n/2$. In either case, from Lemma 2.2, we have:

$$|\Gamma(\mathcal{S}')| \leq \alpha \cdot (1 - \frac{1}{k}) \cdot |\mathcal{S}'|.$$

We know that in G, $|\Gamma(\mathcal{S}')|$ is at least $\alpha \cdot |\mathcal{S}'|$. Hence, the number of faulty nodes in $\mathcal{S}'$s neighborhood must be at least $\alpha(1 - (1 - \frac{1}{k})) \cdot |\mathcal{S}'|$ i.e. greater than $\alpha \cdot \frac{1}{k} \cdot \frac{k \cdot f}{\alpha} = f$. Since the total number of faults allowed to the adversary is at most f, we have a contradiction. Hence, H is at least $n - \frac{k \cdot f}{\alpha}$ in size and has an expansion of at least $(1 - \frac{1}{k}) \cdot \alpha$. □

2.2 Lower bounds for adversarial faults

The result given in Theorem 2.1 is the best possible up to constant factors in a sense that for every $\alpha > 0$ smaller than some constant there is an infinite family of graphs which disintegrate into sublinear components if $f \geq c \cdot \alpha n$ for some sufficiently large constant c.

Theorem 2.3 *There exists a constant γ such that, given any $\alpha < \gamma$, there is an infinite family of graphs with expansion α for which there is an adversarial selection of $c \cdot \alpha \cdot n$ faulty nodes causing the graph to break into sublinear components, where n is the number of nodes in the graph and c is an appropriately chosen constant.*

Proof. To construct this family of graphs let us consider $G(n)$ to be an infinite family of expander graphs with constant expansion γ and constant degree δ.

For each $G \in G(n)$, construct a graph, H, which is a copy of G with each edge replaced by a chain of k nodes (between its two endpoints), where k is even. Then H has $k \cdot (\delta n)/2 + n = \Theta(k \cdot n)$ nodes.

Claim 2.4 *Graph H has expansion $\Theta(\frac{1}{k})$.*

Proof. Take any subset U of nodes in H representing original nodes in G and let U' be the set resulting from U by adding the $k/2$ nearest nodes of each chain a node in U is connected to. Then $|U'| = (\delta \cdot (k/2) + 1) \cdot |U|$ but $|\Gamma(U')| = |\Gamma(U)| \leq \delta \cdot |U|$. Hence,

$$\alpha(U') = \frac{|\Gamma(U')|}{|U'|} \leq \frac{2}{k} \cdot |U'|$$

completing the proof of the claim. □

Now, from each chain of k nodes we remove a central node. Then each connected component remaining has at most $1 + \delta \cdot \frac{k}{2}$ nodes left, i.e. a sublinear number, and the total number of nodes removed is $\frac{\delta}{2} \cdot n$, which is $\frac{1}{k}$ times the number of nodes in the graph. □

We now show a more far-reaching lower bound: For families of graphs that have *uniform expansion*, our upper bound on the maximum number of node faults is not only true in some pathological cases, but for *every* case, i.e. for every graph in every such family of graphs. In the following, whenever we speak about a graph of uniform expansion, we assume that it belongs to a suitable family of graphs.

Theorem 2.5 *For every connected graph of size n and uniform expansion β there is an adversarial selection of $\omega(\beta(n) \cdot n)$ faulty nodes that causes the graph to break into sublinear components.*

Proof. Let $G = (V, E)$ be any graph of uniform expansion β that consists of n nodes. Then there must be a set $U_1 \subseteq V$, $|U_1| \leq n/2$, so that $|\Gamma(U_1)| \leq \beta(n) \cdot |U_1|$. Removing $\Gamma(U_1)$ leaves G with a set $\mathcal{V}_1 = \{V', V''\}$ of two node sets, $V' = U_1$ and $V'' = V \setminus (U_1 \cup \Gamma(U_1))$. Let V_1 be a set in $\mathcal{V}_1$ of maximum size. It follows from the uniformity of G that there must be a set $U_2 \subseteq V_1$, $|U_2| \leq |V_1|/2$, so that $|\Gamma(U_2)|$ w.r.t. $G(V_1)$ is $O(\beta(|V_1|)) \cdot |U_2|$. Removing U_2 results in a new set $\mathcal{V}_2$ of sets of nodes in which V_1 is replaced by U_2 and $V_1 \setminus (U_2 \cup \Gamma(U_2))$. We continue to take a node set V_i of largest size out of $\mathcal{V}_i$ and remove nodes at the minimum expansion part in $G(V_i)$ until there is no subset in $\mathcal{V}_i$ left of size at least ϵn.

Our goal is to show that this process only removes $O(\frac{\log(1/\epsilon)}{\epsilon} \cdot \beta(n) \cdot n)$ nodes from G. If this is true, the theorem would follow immediately. We prove the bound with a charging strategy: Each time a set V_i is selected from $\mathcal{V}_i$, we charge all nodes in $\Gamma(U_{i+1})$ taken away from V_i to the nodes in U_{i+1}. Since

$$|\Gamma(U_{i+1})| = O(\beta(\epsilon n)) \cdot |U_{i+1}| = O\left(\frac{\beta(n)}{\epsilon} \cdot |U_{i+1}|\right)$$

for any $\beta(x) \geq 1/x$, this means that every node in U_{i+1} is charged with a value of $O(\epsilon^{-1} \cdot \beta(n))$. Every node can be charged at most $\log(1/\epsilon)$ times because each time a node is charged, it ends up in a node set U_{i+1} that is at most half as large as V_i, and we stop splitting a node set once it is of size less than ϵn. Hence, at the end, every node in V is charged with a value of $O(\frac{\log(1/\epsilon)}{\epsilon} \cdot \beta(n))$. Summing up over all nodes, the total charge is

$$O\left(\frac{\log(1/\epsilon)}{\epsilon} \cdot \beta(n) \cdot n\right) ,$$

which represents the number of nodes that have been removed from the graph. □

The proof of this theorem indicates that families of graphs with uniform expansion may have a smooth degradation in the size of their largest connected component. This is in fact true for the family of $n \times n$-meshes. However, since there are also families of graphs with a threshold behavior, like the graphs used in Theorem 2.3, Theorem 2.1 is, up to constant factors, the best one can show in general.

2.3 Algorithmic pruning

The pruning step is crucial to the results discussed in this section and it will be central to the handling of random faults in the next section. Before proceeding to that section we discuss briefly the possibility of actually performing a pruning step in polynomial time.

The pruning algorithm can be seen as a solution to the decision version of the sparsest cut problem which can be stated as follows.

Sparsest Cut:
Given a graph $G = (V, E)$ with arbitrary non-negative edge capacities $c(e)$, $e \in E$, and some $\beta > 0$, find a set $U \subseteq V$ with $|U| \leq |V|/2$ and

$$\sum_{u \in U, v \in V \setminus U} c(u, v) < \beta \cdot |U|.$$

The sparsest cut problem is known to be NP hard. However if there were to exist a γ-approximation algorithm for the optimization version of the sparsest cut problem we could use it in the following way for the pruning:

- Find the γ-approximate sparsest cut of the faulty graph.
- If the algorithm finds a set S with $\alpha(S) \leq \epsilon \cdot \alpha$ for some small constant $\epsilon > 0$, prune S away otherwise stop.

Since the expansion of the cut found by the algorithm is at most γ times the expansion of the faulty graph, we know that the expansion of the remaining graph is $\Omega(\alpha/\gamma)$.

The proof that the size of this component is at least a constant fraction of the original size of the graph when allowing $c \cdot \alpha n$ faults for a sufficiently small constant c follows from the existential proof above.

Recently the best known approximation to the sparsest cut problem was improved to $\sqrt{\log n}$ by Arora, Rao and Vazirani [4]. They also conjecture that the sparsest cut problem should be approximable in polynomial time to within a constant factor. This would have the effect of turning the existential results in this paper into algorithmic ones, making it possible to actually find in polynomial time the large component of good expansion whose existence we proved here.

3 Random faults

We now direct our attention to the case of random faults. We assume that each node in the graph can become faulty independently with a given probability p.

3.1 Random faults are not always easier to handle

Intuitively it appears that random faults are easier to handle than adversarial faults, i.e. a graph can survive much more random faults than adversarial faults. However, as we will see in this section, this is not true in general. In fact, there are families of graphs for which a fault probability of $\Theta(\alpha)$ causes the graph to disintegrate into sublinear components, where α is the expansion of the graph. In other words, in these graphs $\Theta(\alpha n)$ random node failures can be as bad as adversarial node failures.

Theorem 3.1 *There exists a constant γ such that, given any $\alpha < \gamma$, there is an infinite family of graphs with expansion $\Theta(\alpha)$ for which a fault probability of $\Theta(\alpha)$ causes the graph to disintegrate into sublinear components, with high probability.*

Proof. We use the family of graphs constructed in the proof of Theorem 2.3, i.e. let $\mathcal{G}$ be an infinite family of constant degree expander graphs with constant expansion γ and constant degree δ. Further, let H be the graph resulting from G by replacing every edge in G by a chain of k nodes (between its two endpoints). Graph H has $\Theta(k \cdot n)$ nodes. From Claim 2.4 we know that H has an expansion of $\Theta(\frac{1}{k})$. Also, H has the following important property.

Claim 3.2 *The number of connected subgraphs H' of H with the property that H' contains exactly r nodes from G and all border nodes in H' (i.e. nodes with an edge leaving H') are nodes in G is at most $n \cdot \delta^{2(r-1)}$.*

Proof. It suffices to count all possible connected subsets of nodes of size r in G. Certainly, any connected subset of nodes of size r can be spanned by a tree with $r-1$ edges. This tree can be traversed by an Eulerian tour in which each edge is used at most twice. Hence, all nodes of the subset can be visited by a walk along at most $2(r-1)$ edges in G. Since the starting point of the walk can be any one of the n vertices, there are at most $n \cdot \delta^{2(r-1)}$ ways of generating a connected subset of size r. □

Let the failure probability of the nodes in H be $p = (2\ln\delta + 2)/k$. Consider any subgraph H' of H with the properties of Claim 3.2 and let $r = \ln n + 1$. Since the nodes of H' can be spanned by a tree of exactly $r-1$ chains, the number of nodes in H is at least $(k+1)(r-1)$. Hence, this particular subgraph survives in H with probability at most $(1-p)^{k(r-1)} \leq e^{-k(r-1)\cdot p}$. By Claim 3.2 there are no more than $n \cdot \delta^{2r}$ such subgraphs in H. Hence, the probability that any such subgraph survives is at most $n \cdot \delta^{2(r-1)} \cdot e^{-k(r-1)\cdot p} = n^{1-2} = 1/n$. But if no such subgraph survives, then a connected component in H can have a size of at most $O(\delta \cdot k \ln n)$. Thus, H breaks down into sublinear components with high probability. □

However, the expansion of a graph is *not* a critical point for all graphs. There are several important classes of graphs which can sustain a much higher fault probability and still yield a linear sized connected component with good expansion.

3.2 Extracting a large subgraph from a graph with random faults

Consider any graph $G = (V, E)$ with n nodes, maximum degree δ, expansion α, and span σ. Let $G_f = (V, E_f)$ be the faulty version of G where an edge $\{u, v\} \in E$ remains in E_f if and only if both u and v are non-faulty. We want to find a graph $H \subseteq G_f$ of size $\Theta(n)$ with expansion $\Omega(\alpha/\delta)$. To obtain a better bound than what we showed for adversarial faults we will deploy the concept of compact sets in this section.

Let $\mathcal{U}$ be the set of all compact sets of G. Recall that a set is compact if it and its complement are connected. In the following, given any graph G', $V(G')$ denotes the node set of G', $\Gamma_{G'}(U)$ denotes the neighborhood of node set U with respect to G' and $\alpha_{G'}(U)$ denotes the expansion of node set U with respect to G'.

In this section instead of simply pruning sets of bad expansion we will be pruning compact sets of bad expansion. The advantage we gain by this is technical and can be simply stated as follows: it is easier to count the number of compact sets with a boundary of a given size. Specifically consider compact sets with boundaries of size k. The proof of Claim 3.2 implies that there are at most $n \cdot \delta^{2\sigma \cdot k}$ ways of generating a tree of size $\sigma \cdot k$ in G. Therefore, there are at most $n \cdot (e\sigma)^k \cdot \delta^{2\sigma \cdot k} \leq n \cdot \delta^{3\sigma \cdot k}$ compact sets with k nodes in their neighborhood.

The following lemma guarantees that our new pruning algorithm, called *Prune2* (see Figure 2), will find suitable candidates at every instance by showing that whenever a graph has a subset with bad expansion, it has a compact subset whose expansion is not much better.

Lemma 3.3 *Consider any two graphs $G = (V, E)$ and $G' = (V, E')$ where G is connected and has a maximum degree of δ and $E' \subseteq E$. If $S \subset V$ is connected in G and $|S| \leq |V|/2$ then there exists a compact set $K_{G'}(S)$ in G with $|K_{G'}(S)| \leq |V|/2$ whose expansion in G' is at most $\delta \cdot \alpha_{G'}(S)$.*

Proof. If $S \in \mathcal{U}$ then $K_{G'}(S)$ is simply S. If $S \notin \mathcal{U}$ then $G \setminus S$ is not connected. Let $\mathcal{C}(S)$ be the set of maximal connected subgraphs of $G \setminus S$. That is, there is no edge between any two sets in $\mathcal{C}(S)$. Because G is connected, this implies that every $C \in \mathcal{C}(S)$ must have an edge into S, and hence $\mathcal{C}(S) \subset \mathcal{U}$. We consider two cases.
Case 1: There is a $C \in \mathcal{C}(S)$ with $|C| \geq |V|/2$.
Then $G \setminus C \in \mathcal{U}$, $S \subseteq G \setminus C$, $|G \setminus C| \leq |V|/2$, and $\Gamma_{G'}(G \setminus C) \subseteq \Gamma_{G'}(S)$. Hence, $G \setminus C$ has an expansion in G' of at most $\alpha_{G'}(S)$. So, $K_{G'}(S) = G \setminus C$.
Case 2: For all $C \in \mathcal{C}(S)$, $|C| < |V|/2$.
If any of the connected components in $\mathcal{C}(S)$ has an expansion of at most $\delta \cdot \alpha_{G'}(S)$ in G' then let that component be $K_G(S)$. If not, then all components $C_i \in \mathcal{C}(S)$ have an expansion strictly larger than $\delta \cdot \alpha_{G'}(S)$, i.e. for all i, $|\Gamma_{G'}(C_i)|/|C_i| > \delta \cdot |\Gamma_{G'}(S)|/|S|$. This implies that also

$$\frac{|\Gamma_{G'}(\cup_i C_i)|}{|\cup_i C_i|} > \delta \cdot \frac{|\Gamma_{G'}(S)|}{|S|}$$

But, $|\Gamma_{G'}(\cup_i C_i)| \leq \delta \cdot |\Gamma_{G'}(S)|$ and $\cup_i C_i = G \setminus S$. Hence, $|S| > |G \setminus S|$, which is a contradiction. Therefore, one of the C_i's must have an expansion of at most $\delta \cdot \alpha_{G'}(S)$. □

Algorithm *Prune2(ϵ)*

```
1: G_0 ← G_f; i ← 0
2: while ∃S_i ⊆ V(G_i) s.t. |Γ_{G_i}(S_i)| ≤ (ε/δ)·α|S_i| and |S_i| ≤ |G_i|/2 and S_i is connected
3:     K_i ← K_{G_i}(S_i)
4:     G_{i+1} ← G_i \ K_i
5:     i ← i + 1
6: end while
7: H ← G_i
```

Figure 2: The pruning algorithm

We use the notation from algorithm *Prune2* in the proof and statement of Theorem 3.4. $K_{G_i}(S_i)$ is defined like $K_{G'}(S)$ in Lemma 3.3 where $G = G(V(G_i))$ is the graph that contains all nodes in G_i and all edges in G that have both endpoints in G_i, and $G' = G_i$. As we will see, $G(V(G_i))$ is guaranteed to be connected for every i, and therefore $K_{G_i}(S_i)$ is well-defined.

Theorem 3.4 *Consider any graph G with degree δ and expansion $\alpha \geq (\gamma\delta \ln^3 n)/n$ for some sufficiently large constant γ and $|\Gamma(U)| \geq \log_\delta |U|$ for every node set U in G. Then* Prune2(ϵ) *returns a graph $H \subseteq G$ of size $|H| \geq n/2$ with expansion at least $(\epsilon/\delta) \cdot \alpha$ with high probability, provided that for the fault probability we have $p \leq 1/(16e \cdot \delta^{8\sigma})$ and for the degradation in expansion we have $\epsilon \leq 1/4$.*

Proof. We begin by showing that the regions pruned are compact. Formally, let $\mathcal{T} = G \setminus H$. Hence $\mathcal{T}$ is the union of all the culled regions. To prove the result we will show that with high probability the size of $\mathcal{T}$ is not more than $n/2$. Let $\{T_1, T_2, \ldots, T_\ell\}$ be the maximal connected components of $\mathcal{T}$ in G.

Claim 3.5 *For every $T_j \in \mathcal{T}$, T_j is compact in G.*

Proof. First of all, notice that $G(V(G_i))$ is connected for every i. This is true because only sets that are compact in $G(V(G_i))$ are removed from G_i for every i, and $G(V(G_0)) = G$. Hence, every compact set K_i pruned away from G_i has at least one edge into $G_i \setminus K_i$ in $G(V(G_i))$. Second, every K_i must be completely contained in one of the T_j. This holds because the T_j's are maximal connected components. Thus, if a K_i were distributed among at least two T_j's, then it cannot be connected, violating the compactness property.

Consider now any set T_j. Because it is a maximal connected component, it has no edges to any T_k with $k \neq j$. But T_j must represent the union of compact sets that were culled away, and the last of these that was culled away must

have at least one edge into $G \setminus T_j$. Hence, T_j must have at least one edge into H for every j. Thus, $G \setminus T_j$ is connected for every T_j implying that every T_j is compact in G. □

We now establish a bound on the probability that a given set T_i gets culled. This bound is in terms of δ, σ and the size of the boundary of T_i.

In the following, let $\Gamma(U)$ denote the neighborhood of U in G and $\Gamma_f(U)$ denote the neighborhood of U in G_f. From the definition of α it follows that $|\Gamma(T_i)| \geq \alpha|T_i|$ for every i. Furthermore, it follows from Lemma 3.3 that

$$|\Gamma_f(T_i)| \leq \sum_{K_j \in T_i} |\Gamma_{G_j}(K_j)| \leq \sum_{K_j \in T_i} \epsilon\alpha|K_j| = \epsilon\alpha|T_i| \ .$$

A set T_i is part of the culled region in the even that that it's neighborhood goes down from $\Gamma(T_i)$ to $\Gamma_f(T_i)$. This probability is at most

$$\begin{aligned} \binom{|\Gamma(T_i)|}{|\Gamma_f(T_i)|} \cdot p^{|\Gamma(T_i)|-|\Gamma_f(T_i)|} &\leq \left(\frac{ep|\Gamma(T_i)|}{|\Gamma(T_i)| - |\Gamma_f(T_i)|}\right)^{|\Gamma(T_i)|-|\Gamma_f(T_i)|} &(2)\\ &\leq \left(\frac{ep}{1-\epsilon}\right)^{(1-\epsilon)|\Gamma(T_i)|} &(3)\end{aligned}$$

Note that this is valid under the condition that $ep < 1 - \epsilon$. If we set

$$\epsilon \leq \frac{1}{4} \quad \text{and} \quad ep \leq \frac{1}{16\ \delta^{8\sigma}}$$

then it follows that

$$\Pr[T_i \text{ is culled by } \textit{Prune2}] \leq (1/8) \cdot \delta^{-6\sigma|\Gamma(T_i)|} \ .$$

In order to establish that the probability of the sizes of the T_is cumulatively adding up to more than $n/2$ is low we need to enumerate two cases on the size of the neighborhood of the T_i's.

In case 1 we argue that a T_i with a neighborhood of more than k nodes is unlikely with high probability, where $k = \lceil \log_\delta n \rceil$.

Case 1: There exists an i with $|\Gamma(T_i)| > k$.

We know from above that the probability that a given compact subgraph T_i is culled is at most $\delta^{-6\sigma|\Gamma(T_i)|}$. We multiply this probability with the number of ways of choosing such a subgraph. This gives us the probability that there is a T_i with such a large neighborhood. According to the definition of σ, a neighbor set of $|\Gamma(T_i)|$ nodes can be spanned by a tree with at most $\sigma \cdot |\Gamma(T_i)|$ nodes. Every tree with $\sigma \cdot |\Gamma(T_i)|$ nodes can span at most $\binom{\sigma\cdot|\Gamma(T_i)|}{|\Gamma(T_i)|}$ node sets of size $|\Gamma(T_i)|$. The proof of Claim 3.2 implies that there are at most $n \cdot \delta^{2\sigma\cdot|\Gamma(T_i)|}$ ways of generating a tree of size $\sigma \cdot |\Gamma(T_i)|$ in G. Therefore, there are at most $n \cdot (e\sigma)^{|\Gamma(T_i)|} \cdot \delta^{2\sigma\cdot|\Gamma(T_i)|} \leq n \cdot \delta^{3\sigma\cdot|\Gamma(T_i)|}$ compact sets with $|\Gamma(T_i)|$ nodes in the neighborhood. Note that by definition, $\sigma \geq 1$. Hence,

$$\begin{aligned} \Pr[\exists T_i : \ |\Gamma(T_i)| > k] \quad &\leq \textstyle\sum_{t=k}^{n} n \cdot \delta^{3\sigma\cdot t} \cdot \delta^{-6\sigma\cdot t} &(4)\\ &\leq n^2 \cdot \delta^{-3\sigma k} \leq \tfrac{1}{n} \ . &(5)\end{aligned}$$

Case 2: For all i, $|\Gamma(T_i)| \leq k$.

In case 2 we show that if all T_i's have a neighborhood of at most k, then it is unlikely that $\Sigma_i|T_i|$ is more than $n/2$ with high probability. This is done by looking at the sets $\mathcal{S}_t$, $t \in \{1, \ldots, k\}$, of all possible candidates for the T_i's with t neighbors. For technical reasons we have to count separately for those t in which it is likely that we have $\Omega(\log^2 n)$ candidates: we show that the total number of the nodes in these candidates is bounded by $n/4$, with high probability. We also show the same result for the remaining values of t, adding these up with the earlier nodes to a total of at most $n/2$ nodes that can be culled, with high probability.

Let $\mathcal{S}_t$ be the set of all compact sets in G with a neighborhood of size t. We know from above that $|\mathcal{S}_t| \leq n \cdot \delta^{3\sigma t}$. A subset $\mathcal{S}'_t$ of $\mathcal{S}_t$ is called a *valid candidate* for *Prune2(ϵ)* if and only if for any two sets $T, T' \in \mathcal{S}'_t$, $(T \cup \Gamma(T)) \cap T' = \emptyset$ and $T \cap (T' \cup \Gamma(T')) = \emptyset$. This is because only then T and T' are maximal connected components. A set $T \in \mathcal{S}_t$ is called *bad* if $|\Gamma_f(T)| \leq \epsilon \cdot \alpha|T|$. Certainly, if *Prune2(ϵ)* culls s compact sets T with $|\Gamma(T)| = t$, then there must exist

a valid candidate $\mathcal{S}'_t$ in $\mathcal{S}_t$ of size at least s in which all sets are bad. But if there exists a valid candidate $\mathcal{S}'_t$ in $\mathcal{S}_t$ of size at least k in which all sets are bad, then there must exist a set $\mathcal{S}''_t$ of size at least $k/(\delta t)$ in $\mathcal{S}_t$ with the property that all sets in $\mathcal{S}''_t$ are bad and for all sets $T, T' \in \mathcal{S}''_t$ it holds that $(T \cup \Gamma(T)) \cap (T' \cup \Gamma(T')) = \emptyset$. (A set $\mathcal{S}''_t$ fulfilling the latter property is also called an *independent set*.) This holds because for every node, a valid candidate $\mathcal{S}'_t$ can have at most δ sets that have this node in their neighborhood, and therefore a set in $\mathcal{S}'_t$ can share its neighborhood with at most $(\delta - 1)t \leq \delta t - 1$ other sets in $\mathcal{S}'_t$.

Independent sets of compact sets have the advantage that the probabilities of the compact sets being bad are independent. So our goal will be to bound the probability that there exists an independent set $\mathcal{S}''_t$ in $\mathcal{S}_t$ of size at least s in order to bound the number of compact sets with neighborhood t that *Prune2(ϵ)* may cull.

In order to bound the maximum size of an independent set in $\mathcal{S}_t$, let us simply assume that all sets in $\mathcal{S}_t$ are independent. We are certainly overestimating the maximum size of an independent set in this way because not every collection of sets in $\mathcal{S}_t$ fulfills the independence assumption. Let the random variable X represent the number of bad sets in $\mathcal{S}_t$. Since the probability that a set $T \in \mathcal{S}_t$ is bad is at most $(1/8) \cdot \delta^{-6\sigma t}$, it follows that

$$\mathrm{E}[X] \leq n \cdot \delta^{3\sigma t} \cdot (1/8) \cdot \delta^{-6\sigma t} = n/(8\delta^{3\sigma t}) \; .$$

Furthermore, it follows from the Chernoff bounds that for $n/(8\delta^{3\sigma t}) \geq 6 \ln n$,

$$\Pr[X \geq n/(4\delta^{3\sigma t})] \leq e^{-n/(3 \cdot 8\delta^{3\sigma t})} \leq e^{-2 \ln n} = \frac{1}{n^2}$$

with high probability. Hence, for these values of t, the maximum size of an independent set in $\mathcal{S}_t$ is bounded by $n/(4\delta^{3\sigma t})$, with high probability. From the previous arguments this implies that the maximum number of sets with neighborhood t that *Prune2(ϵ)* can cull is at most $(\delta t) \cdot n/(4\delta^{3\sigma t})$, with high probability. Because we assumed that $|\Gamma(U)| \geq \log_\delta |U|$ for every node set U in G, it further holds that

$$|T| \leq \min \left[\delta^{|\Gamma(T)|}, \; \alpha^{-1} |\Gamma(T)| \right] \; .$$

Hence, for any t so that $n/(8\delta^{3\sigma t}) \geq 6 \ln n$, *Prune2(ϵ)* can cull at most

$$m_t = \delta^t \cdot (\delta t) \cdot n/(4\delta^{3\sigma t}) \leq n/(4\delta^{\sigma t})$$

nodes, with high probability.

Let t_0 be the lowest t so that $n/(8\delta^{3\sigma t}) < 6 \ln n$ (i.e. the bounds above do not apply). Then we define the random variable X as the number of bad sets in $\cup_{t=t_0}^{k} \mathcal{S}_t$, assuming (as a worst case) that all sets are independent. In this case,

$$\mathrm{E}[X] \leq \sum_{t=t_0}^{k} n \cdot \delta^{3\sigma t} \cdot (1/8) \cdot \delta^{-6\sigma t} = \sum_{t=t_0}^{k} n/(8\delta^{3\sigma t}) \leq n/(4\delta^{3\sigma t_0})$$

and the Chernoff bounds imply that

$$\Pr[X \geq 24 \ln n] \leq e^{-(12 \ln n)/3} = \frac{1}{n^4} \; .$$

Hence, *Prune2(ϵ)* can cull at most

$$m_t = \alpha^{-1} k \cdot (\delta k) \cdot 24 \ln n \leq n/4$$

nodes in $\cup_{t=t_0}^{k} \mathcal{S}_t$, with high probability, if $\alpha \geq (96 \; \delta \ln^3 n)/n$. Thus, overall, *Prune2(ϵ)* can cull at most

$$\left(\sum_{t=1}^{t_0} m_t \right) + n/4 = \left(\sum_{t=1}^{t_0} n/(4\delta^{\sigma t}) \right) + n/4 \leq n/4 + n/4 = n/2$$

nodes in case 2, with high probability, which implies that H must have have at least $n/2$ nodes, with high probability.

Combining the two cases, we get

$$\Pr[\text{nodes pruned} \geq n/2] \leq \Pr[\text{Case 1}] + \Pr[\text{Case 2}] \leq \frac{2}{n}$$

which completes the proof. □

3.3 Span of the mesh

Theorem 3.6 *The d-dimensional mesh has span 2.*

Proof. Consider a compact set S in the d-dimensional mesh M. Let B be the boundary nodes $\Gamma(S)$. We place virtual edges between nodes in B. Two distinct nodes $u = (u_0, \ldots u_{d-1})$ and $v = (v_0, \ldots v_{d-1})$ have a virtual edge between them if $|v_i - u_i| = 0$ for at least $d-2$ of its dimensions and $|v_i - u_i| \leq 1$ for the rest. Call the set of such virtual edges E_v. In Lemma 3.7, stated below, we claim that the graph (B, E_v) is connected. Therefore, we can find a spanning tree for B which has exactly $|B| - 1$ virtual edges. Since each edge in E_v can be simulated by exactly 2 edges of M, we can say that there is a spanning tree in M for the nodes of B with at most $2 \cdot (|B| - 1)$ edges. □

Lemma 3.7 *Let $S \subset Z^d$ be a finite compact set, let B be the boundary nodes $\Gamma(S)$, and let E_v be the set of virtual edges. Then the graph (B, E_v) is connected.*

Proof. We will show that for any two points u and v in B, there is a path in E_v connecting the two; if this can be done for every two points, then B is connected as we hope to prove.

Our proof uses some basic and standard homology theory of cell complexes, which can be found in any introductory topology text; for instance, see [14]. Specifically, we use the Z_2 homology of d-dimensional Euclidean space R^d. We partition R^d into a complex of unit hypercube cells having the points of Z^d as their vertices. Each d-dimensional unit hypercube cell has as its boundary a set of $2d$ $(d-1)$-dimensional unit hypercube facets, again having Z^d as vertices, and so on. In this complex, a *k-chain* is defined to be any finite set of k-dimensional unit hypercubes having points of Z^d as vertices. The boundary of a k-chain C is the symmetric difference of the boundaries of its hypercubes; that is, it is the set of $(k-1)$-dimensional hypercubes that are on the boundary of an odd number of the k-dimensional hypercubes in C. A *k-cycle* is defined to be a k-chain that has an empty boundary, and a *k-boundary* is defined to be a k-chain that is the boundary of some $(k+1)$-chain. For quite general classes of cell complexes in more complicated topological spaces than R^d, every k-boundary is a k-cycle, but in R^d, the reverse is also known to be true: every k-cycle is a k-boundary.

Now, given u and v, since S is connected we can find a path p_1 connecting u to v by a sequence of adjacent points in S. We also find an edge e_1 connecting u to an adjacent point of Z^d outside S, an edge e_2 connecting v to an adjacent point of Z^d outside S, and a path p_2 connecting these two exterior points by a sequence of adjacent points outside S (since the complement of S is connected). The union of p_1, p_2, and $\{e_1, e_2\}$ forms a 1-chain in the cubical complex described above. Moreover, this is a 1-cycle, because it has degree two at every vertex it touches. Therefore, it is the boundary of a 2-chain C; that is, C is a set of squares and $p_1 \cup p_2 \cup \{e_1, e_2\}$ is the set of edges in the cubical complex that touch odd numbers of squares in C.

Next, let U be the subset of R^d formed by a union of axis-aligned unit hypercubes, one for each member of S, and having that member as its centroid; note that these hypercubes do not have integer vertices. Let B be the $(d-1)$-boundary of U; B consists of a collection of $(d-1)$-dimensional unit hypercubes that again do not have integer vertices. Finally, let $G = B \cap C$.

To understand the structure of G, consider the intersection of a square s of C and a $(d-1)$-dimensional hypercube h of B. Observe that: (1) all points in s have at least $(d-2)$ integer coordinates values, (2) the locus of points in h that have at least $(d-2)$ integer coordinates values is a set of $(d-1)$ line segments, and (3) if s intersects h, there are two adjacent vertices of s such that one is in S and the other in the complement of S. Thus, if s and h meet, they do so in a line segment of length $1/2$, that connects the centroid of h (where it is crossed by one edge of the square) to the centroid of one of its boundary $(d-2)$-dimensional hypercubes. Therefore G, the union of these line segments, can be viewed as a graph that connects vertices at these points. The degree of a vertex at the centroid of h is equal to the number of squares of C that touch that point, and the degree of the other vertices can only be two or four depending on which of the four vertices of the square defining the vertex is interior to U.

Since the boundary of C crosses B only on the two edges e_1 and e_2, these two crossing points have odd degree and all the other vertices of G have even degree. Any connected component of any graph must have an even number of odd-degree vertices, so the two odd vertices $e_1 \cap B$ and $e_2 \cap B$ must belong to the same component and can be connected by a path p_3 in G.

Each length-$1/2$ segment of p_3 belongs to the boundary of a single hypercube in U, which has as its centroid a point of B. Let p_4 be the sequence of centroids corresponding to the sequence of edges in p_3. Then p_4 starts at u, and ends at v. Further, at each step from one edge in p_4, either the current point in B does not change, or it changes from one point in B to an adjacent point (when the corresponding pair of edges in p_4 form a 180° angle on two adjoining

hypercubes), or it changes from one point in B to a point at distance $\sqrt{2}$ away (when the corresponding edges in p_4 form a 270° angle across a concavity on the boundary of U).

So, we have constructed a path in E_v between an arbitrarily chosen pair of points u, v in B, and therefore the graph (B, E_v) is connected. □

Theorem 3.6 implies that the d-dimensional mesh can sustain a fault probability inversely polynomial in d and still have a large component whose expansion is no more than a factor of d worse than the original. In the next section we elaborate on the significance of this result.

4 Conclusion

In this paper we presented a general technique for determining the robustness of the expansion of different graphs against both adversarial and random faults. For random faults we have come up with a new parameter, the span, which allows us to prove a strong result regarding the robustness of high dimensional meshes. Among other things, this result can provide useful insights into the working of peer-to-peer networks like CAN [25] which behaves like a d-dimensional mesh in its steady state. Basically we have shown that CAN can tolerate a fault probability which is inversely polynomial in its dimension without losing too much in its expansion properties.

For the 2-dimensional mesh our result is related to the line of research followed by Raghavan [24], Kaklamanis et. al. [15] and Mathies [23] who show that despite a constant fault probability (of as high as 0.4) a mesh with random failures can emulate a fault free mesh using paths with stretch factor at most $O(\log n)$. Since the distance of nodes in a graph of expansion α is $O(\alpha^{-1} \log n)$ [21], our technique gives essentially the same result albeit with a lower fault probability. Additionally, for meshes of constant dimension greater than 2 our results imply a $O(\log n)$ dilation for path lengths, and hence a way to generalize these earlier results to higher dimensions.

The strength of our technique is that it is able to yield results for the 2-dimensional mesh which are comparable to previous results while giving new results for higher dimensional meshes and providing a general method suitable for analyzing any network whose span can be estimated.

Open problems

We conjecture that the butterfly, shuffle-exchange, and deBruijn graph all have a span of $O(1)$, which means that they can tolerate a constant fault probability. Though the span may provide tight results for these graphs, the exponential dependency of the fault probability on the span does not really give useful results if the span is beyond $\log n$. Hence, either a better dependency result is needed or a parameter better than the span is needed. Clearly, as mentioned in the introduction, having a parameter that can accurately describe the fault tolerance of graphs w.r.t. expansion under random faults would be very useful for many applications.

References

[1] M. Ajtai, J. Komlós, and E. Szemerédi. Largest random component of a k-cube. *Combinatorica*, 2(1):1–7, 1982.

[2] Noga Alon and F. R. K. Chung. Explicit construction of linear sized tolerant networks. *Discrete Math*, 72:15–19, 1989.

[3] E. Anshelevich, D. Kempe, and J. Kleinberg. Stability of load balancing algorithms in dynamic adversarial systems. In *Proc. of the 34th ACM Symp. on Theory of Computing (STOC)*, 2002.

[4] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *Proc. of 36th Annual ACM Symp. on Theory of Computing*, pages 222–231, 2004.

[5] M. Ben-Or and D. Ron. Agreement in the presence of faults, on networks of bounded degree. *Information Processing Letters*, 57(6), 1996.

[6] B. Bollobas. The evolution of random graphs. *Transactions of the AMS*, 286:257–274, 1984.

[7] R. Cole, B. Maggs, and R. Sitaraman. Routing on butterfly networks with random faults. In *Proc. of the 36th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 558–570, 1995.

[8] Richard Cole, Bruce M. Maggs, and Ramesh K. Sitaraman. Multi-scale self emulation: a technique for reconfiguring arrays with faults. In *Proc. of the 25th ACM Symp. on Theory of Computing (STOC)*, pages 561–572, 1993.

[9] R.J. Cole, B.M. Maggs, and R.K. Sitaraman. Reconfiguring arrays with faults part I: worst-case faults. *SIAM Journal on Computing*, 26(6):1581–1611, 1997.

[10] C. Dwork, D. Peleg, N. Pippenger, and E. Upfal. Fault tolerance in networks of bounded degree. *SIAM Journal on Computing*, 17:975–988, 1988.

[11] P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.*, 5:17–61, 1960.

[12] Bhaskar Ghosh, Frank T. Leighton, Bruce M. Maggs, S. Muthukrishnan, C. Greg Plaxton, Rajmohan Rajaraman, A. W. Richa, R. E. Tarjan, and David Zuckermann. Tight analyses of two local load balancing algorithms. *SIAM Journal on Computing*, 29(1):29–64, 1999.

[13] J. Håstad, T. Leighton, and M. Newman. Reconfiguring a hypercube in the presence of faults. In *Proc. of the 19th ACM Symp. on Theory of Computing (STOC)*, pages 274–284, 1987.

[14] M. Henle. *A Combinatorial Introduction to Topology*. W. H. Freeman, San Francisco, CA, 1979.

[15] C. Kaklamanis, A. R. Karlin, F. T. Leighton, V. Milenkovic, P. Raghavan, S. Rao, and A. Tsantilas. Asymptotically tight bounds for computing with faulty arrays of processors. In *Proc. of the 22nd ACM Symp. on Theory of Computing (STOC)*, pages 285–296, 1990.

[16] A. Karlin, G. Nelson, and H. Tamaki. On the fault tolerance of the butterfly. In *Proc. of the 26th ACM Symp. on Theory of Computing (STOC)*, pages 125–133, 1994.

[17] H. Kesten. The critical probability of bond percolation on the square lattice equals $1/2$. *Communication in Mathematical Physics*, 74:41–59, 1980.

[18] F.T. Leighton and B.M. Maggs. Fast algorithms for routing around faults in multibutterflies and randomly-wired splitter networks. *IEEE Transactions on Computers*, 41(5):578–587, 1992.

[19] F.T. Leighton, B.M. Maggs, and S.B. Rao. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Combinatorica*, 14(2):167–186, 1994.

[20] F.T. Leighton, B.M. Maggs, and R.K. Sitaraman. On the fault tolerance of some popular bounded-degree networks. *SIAM Journal on Computing*, 27(5):1303–1333, 1998.

[21] Tom Leighton and Satish Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *29th Annual Symposium on Foundations of Computer Science*, pages 422–431, 1988.

[22] T. Luczak, B. Pittel, and J.C. Wierman. The structure of a random graph at the point of the phase transition. *Transactions of the AMS*, 341:721–748, 1994.

[23] T. R. Mathies. Percolation theory and computing with faulty arrays of processors. In *Proc. of the 3rd ACM Symp. on Discrete Algorithms (SODA)*, pages 100–103, 1992.

[24] Prabhakar Raghavan. Robust algorithms for packet routing in a mesh. In *Proc. of the 1st ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 344–350, 1989.

[25] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard M. Karp, and Scott Schenker. A scalable content-addressable network. In *Proc. of SIGCOMM 2001*, pages 161–172, 2001.

[26] C. Scheideler. *Universal Routing Strategies for Interconnection Networks*. Lecture Notes in Computer Science 1390. Spinger, 1998.

[27] Christian Scheideler. Models and techniques for communication in dynamic networks. In *Proc. of the 19th Symp. on Theoretical Aspects of Computer Science (STACS)*, pages 27–49, 2002.

[28] E. Upfal. Tolerating a linear number of faults in networks of bounded degree. *Information and Computation*, 115(2):312–320, 1994.