Noise Tolerance of Expanders and Sublinear Expansion Reconstruction

Satyen Kale Yuval Peres
Microsoft Research
One Microsoft Way
Redmond, WA 98052
{satyen.kale, peres}@microsoft.com

C. Seshadhri
Dept. of Computer Science,
Princeton University
35 Olden St, Princeton, NJ 08540
csesha@cs.princeton.edu

November 7, 2009

1 Introduction

Large data sets are ubiquitous today with the advent of high speed computers and connectivity. Being able to gainfully use these data sets, however, is a challenge, especially when they are used to answer online queries which are generated rapidly. Even linear time algorithms are too slow in this context, and therefore sublinear time algorithms become critical.

In many applications, queries to large data sets typically assume expect the data set to satisfy some structural property that makes it useful. For example, users of a data set may expect that a set of points is in convex position, a list of numbers is sorted, or a large graph is a tree. These properties are sensitive to noise, and even a small amount of corruption could destroy the usefulness of the data set. For example, suppose we performed binary search on an array that is not completely sorted: even a few values out of place could make the binary search process completely fail.

However, because of various difficulties in gathering data on such a large scale, or even simply because of corruption due to external noise, the data set may actually fail to satisfy the desired property. The extensive theory of property testing algorithms (see surveys [12, 13, 22]) provides means to detect such problems in existing data sets in sublinear time.

However, in applications it may not suffice just to detect such problems, after all, one needs to actively process the data to perform useful tasks. In such situations, it is reasonable to minimally modify the data to restore the property, so that further processing can take place without problems. However, since the input is so massive, one cannot afford to read the whole input and then fix it, particularly since this may require global changes. Thus, at first blush, it may seem quite impossible to repair the data set entirely in sublinear time. However, if the data set is so gargantuan that one cannot even afford to read it entirely, then usually it is the case that access to the input is provided in the form of local queries: for example, if the input is, say, the web graph, then typical queries to the input would ask for all the outgoing links from a given webpage.

In such cases, it might be possible to repair the input as and when we read it. More precisely, we seek to design a *filter* algorithm that sits between users of the input and the input itself. The filter accepts queries and actively repairs the input while answering the queries. This was the motivation for the definition of *online property reconstruction* model in [1] (described in more detail later).

In this paper, we are concerned with online sublinear reconstruction of expanders. Informally stated, the problem is the following. The input is a huge bounded degree graph represented by adjacency lists. The graph is supposed to be an *expander*, i.e. for any subset of vertices of at most

half the size of the graph, the number of outgoing edges (i.e., to the complement of the subset) is a significant fraction of the total number of edges incident on the vertices in the set. The queries are in the form of requests for the adjacency list of a specified vertex. The filter's job is to answer such queries in sublinear time, such that the final picture is that of a graph that really is an expander, while adding as few edges as possible to make it into an expander.

We show that expander reconstruction is connected to some intriguing questions about random walks in "noisy" expanders. Suppose one is given a graph G that consists of a large expander connected arbitrarily to some small unknown graph (the noise). The graph G is probably not an expander. Nonetheless, we would like to show that random walks on the expander part are not greatly affected by the noise. Turning this around, we can ask: what portion of the large expander is affected by this noise? Can we still hope that from almost all of the expander, we can reach a vast majority of the expander by short random walks? We prove the rather strong statement that the noise can affect a part of the expander that is proportional in size to the noise. We believe that this result should be useful in other contexts as well, and we present it in the general setting of arbitrary irreducible Markov chains rather than for our specific application.

For the reconstruction problem, we design a sublinear time procedure based on random walks that can very accurately distinguish between vertices in a high expansion part of the graph, and vertices of the noisy part. This procedure essentially measures the distance between the probability distribution induced by a random walk on an undirected graph to the stationary distribution. Previous algorithms for measuring such distances [7] are not efficient enough for our situation, and we design new tools for these scenarios. Using our general result on the noise-tolerance of expanders, we show that the number of vertices from which a random mixes poorly is a good measure of the combinatorial distance (in terms of edges changed) of a graph to being an expander, thus establishing a link between these two different notions of distance.

Our implementation of the filter algorithm allows it to answer queries in *parallel*. This is a very useful property, in settings where we imagine the filter to be a query processing engine for a giant database, and there are many independent users who all want to have access to the database. The ability to answer queries in parallel is certainly more difficult to achieve than sequential answering, when the filter may be able to store a limited amount of history, mainly because the answers of the filter need to be consistent over all the parallel queries.

This paper is organized as follows. In the next few subsections, we formally define the online reconstruction model, present the problem of online sublinear reconstruction of expanders and describe our main result. We also discuss some related work. In Section 2, we describe our results on the noise-tolerance of expanders in a general setting. In Section 3, we describe our reconstruction algorithm, and in Sections 4 and 5 we prove the properties of our reconstruction algorithm. Finally, we present some conclusions and future directions for research.

1.1 The Online Reconstruction Model

We now give a more formal description of online reconstruction as given in [1]. We think of the input as represented as a function f (which we have oracle access to) defined on some appropriate domain. We wish to output function g such that g satisfies to desired property and f is modifed minimally to get g. The function g is output in an *online sublinear* manner by the *filter*: there are a series of queries (in the form of domain elements) $x_1, x_2, \cdots$ made, and the (possibly randomized) filter outputs $g(x_1), g(x_2), \cdots$ where each answer is output in *sublinear* time (in the size of the domain).

The filter is allowed to store previous query answers to help it maintain a consistent output. Viewing the input f as a giant database and the filter as a query processing engine, it is natural to

expect many queries to be posed in parallel, and all answers should be consistent (in that they are outputs of the same function g). Thus, we would like to have a *parallel filter* (introduced in [24]). Since the filter is typically randomized, it needs access to a small sublinear sized random seed s. To allow for a parallel implementation, we fix this seed once and for all. The filter is then a *deterministic* sublinear time procedure F such that $g(x) = F(x, s)$. The function g should satisfy the required property with high probability over the choice of the random seed s, over all possible queries x. The deterministic implementation of F ensures that queries can be answered in parallel.

1.2 Online Sublinear Reconstruction of Expanders

The input is a large graph $G = (V, E)$, with n vertices. We assume that all the vertices have degrees bounded by some specified constant d, which we assume to be sufficiently large (say at least 10). The graph G is represented by *adjacency lists*: for every vertex v, there is list of vertices (of size at most d) adjacent to v. Given a subset of vertices $S \subseteq V$ of size at most $n/2$, let $\bar{S} = V \setminus S$, and let $E(S, \bar{S})$ denote the set of edges crossing the cut $(S, \bar{S})$. The expansion of the cut is defined to be $|E(S, \bar{S})|/|S|$. The expansion of the graph is the minimum expansion over all cuts in the graph. An expander is a generic term for a graph with high expansion.

For reasons that will become obvious later, we consider a related parameter, the conductance of a cut, which is defined to be just the expansion of the cut divided by $2d$. The conductance of the graph is its expansion divided by $2d$. We are given a parameter ϕ, and the input graph is supposed to have the property that its conductance is at least ϕ.

The filter will be represented as the procedure RECONSTRUCT and will, on receiving a query vertex, output its adjacency list in the reconstructed expander graph G'. We assume access to some sublinear size random seed s which is fixed at the beginning before receiving any queries. The deterministic procedure RECONSTRUCT takes a vertex v and the random seed s as inputs and outputs at most d neighbors of v in G' (the adjacency list of v). Each query is answered in sublinear time and the answers are consistent (thereby giving a description of the graph G'). The final graph G' has conductance at least ϕ' for some $\phi' < \phi$, degrees bounded by d, and the number of edges changed (from G to get G') is at most a small factor more than the optimal number of edges that need to be changed to make the conductance at least ϕ. We now state our main theorem regarding the properties of our filter:

Theorem 1 *There is a determinisitic procedure* RECONSTRUCT *that takes as input a vertex v and random seed s of length $\tilde{O}(\sqrt{n})$[1], and outputs the adjacency list of v in G'. The following properties hold with high probability[2]:*

1. *Each query can be answered in parallel in $\tilde{O}(\sqrt{n})$ time,*
2. *All query answers are consistent: the vertex u is output as a neighbor of v iff v is output as a neighbor of u,*
3. *The final graph G' has conductance at least $\phi' = \Omega(\phi^2/\log n)$ and degree bound d,*
4. *The number of edges changed is at most $O(\frac{1}{\phi} OPT)$, where OPT is the optimal number of edges needed to be changed to make the conductance at least ϕ.*

Note that the parallel filter can be used as a property tester for expansion, so by the lower bound for testing expansion by Goldreich and Ron [17], the running time of our filter is optimal upto $\operatorname{poly}\log(n)$ factors.

[1] In this paper, we use the $\tilde{O}$ notation to suppress dependence on $\operatorname{poly}\log(n)$ factors.

[2] The probability of any of the properties failing to hold is at most $1/n^2$.

1.3 Related work

In *property testing* [14, 23], we wish to accept inputs that satisfy some given property, and reject those that are sufficiently "far" from having that property. The significant achievement of the theory of property testing algorithms is showing that a vast array of combinatorial and graph properties [2, 3, 10, 14] are indeed testable without even reading the entire input. The problem of online reconstruction is closesly connected to property testing: indeed, a parallel filter can be used to estimate the distance to a given problem in sublinear time, thereby, giving a property testing (and even a *tolerant tester* [21]).

The input model of adjacency lists for sparse graphs was introduced by Goldreich and Ron [17], where they designed testers for a large set of problems. More general results about the testability of large classes of properties were obtained by Czumaj and Sohler [10] and by Benjamini, Schramm, and Shapira [8]. The technique of using random walks for testing was first introduced by Goldreich and Ron [15] for testing bipartiteness. Goldreich and Ron [16] formally posed the problem of expansion testing (by a result in [17], there is a lower bound of $\Omega(\sqrt{n})$ queries to the input graph). The first progress towards this was made by Czumaj and Sohler [11]. Further work on testing expansion improving certain parameters was done in [18, 20].

The first results for online filtering were obtained for monotonicity testing by Ailon *et al* [1]. These were then extended to parallel filters [24]. There has also been work on online sublinear filters for some computational geometric settings [9].

2 Noise-tolerance of Markov Chains

First, we discuss our theorem about the noise-tolerance of Markov chains. Later, we apply this to our setting of expansion reconstruction. Let M be a finite Markov chain with state set V and transition probabilities p_{uv} for $u, v \in V$. For simplicity, we assume that M is irreducible, and that for any $u \in V$, $p_{uu} \geq 1/2$ (so that the chain is aperiodic). In this case, there is a unique stationary distribution π for the Markov chain. For a subset of states $S \subseteq V$, define $\pi(S) = \sum_{u \in S} \pi_u$. Note that the chain need not be reversible.

The conductance of the Markov chain is defined to be the largest number ϕ such that for any subset of states $S \subseteq V$,

$$\sum_{u \in S, v \in V \setminus S} \pi_u p_{uv} \;\geq\; \phi \cdot \min\{\pi(S), \pi(V \setminus S)\}.$$

It is a well-known fact (see, for example [25]) that if the Markov chain has high conductance, then from any starting state, the chain converges to the stationary distribution exponentially fast at a rate determined by the conductance.

Suppose we are given a Markov chain which *almost* has high conductance, in the following sense. There is a "large" subset of states $V' \subseteq V$ such that the chain restricted to the subset V' has high conductance, and nothing can be said about the remaining "small" part of the state space, $B = V \setminus V'$. This part B is the "noise". In that case, we would like to show that except for a set of states B' of stationary measure comparable to that of B, from all other starting states the chain will have some reasonably nice mixing properties. Intuitively, a small bad part in a Markov chain should not affect the mixing properties of more than, say, twice as many starting states (in the stationary measure).

This turns out to be hard to prove because it can be true for completely different reasons. Suppose that the noise B is almost disconnected from V'. Then, a random walk starting from V'

will almost never leave V' and since V' has high conductance, will definitely mix rapidly inside V'. On other other hand, suppose that the whole chain has high conductance (not just restricted to V'). In this case, although walks from inside V' will encounter B, all walks will still mix rapidly. We need a proof that can interpolate between these scenarios. A first approach to proving this would be to estimate the probability that a walk from V' never hits B. But such a bound would be too weak - there are many states (compared to $\pi(B)$) which are sufficiently "close" to B that a random walk from them will almost certainly hit B.

We now proceed to formalize the setting. Given a subset of states $V' \subseteq V$, define the V'-conductance of the chain to be the largest number ϕ such that for any set $S \subseteq V'$,

$$\sum_{u \in S, v \in V' \setminus S} \pi_u p_{uv} \;\geq\; \phi \cdot \min\{\pi(S), \pi(V' \setminus S)\}.$$

We also need the notion of a *uniform averaging walk* in the Markov chain of t steps: in such a walk, we choose a number $r \in \{0, 1, 2, \ldots, t-1\}$ uniformly at random, and stop the chain after r steps. We will use the notion of *total variation distance* between two distributions p and q on the states:

$$\|p - q\|_{\mathrm{TV}} \;=\; \max_S |p(S) - q(S)| \;=\; \frac{1}{2}\|p - q\|_1,$$

where S is an arbitrary subset of states, and $p(S)$ and $q(S)$ are, respectively, the measures of S under p and q respectively.

Theorem 2 *Let the Markov chain M have V'-conductance ϕ. Let $\pi_0 = \min\{\pi_u/\pi(V') : u \in V'\}$. Then, there is a set B' such that $\pi(B') \leq 2\pi(B)$ with the property that starting from any state $s \in V \setminus B'$, if the uniform averaging Markov chain of $t \geq 8\log(2/\epsilon\pi_0)/(\epsilon\phi^2)$ steps is run, then the final probability distribution $\vec{p}_s$ satisfies*

$$\|\vec{p}_s - \pi\|_{TV} \;\leq\; \epsilon + \pi(B).$$

We prove this theorem as follows. We consider a closely related Markov chain that is restricted to V'. We retain all the original transitions in M between any pair of vertices $u, v \in V'$ with the same probabilities. We assign all such transitions a cost of 1. The meaning of this cost will be explained later.

For any pair of vertices $u, v \in V'$, and for every integer $t \geq 2$, we add a new transition $e_{uv}^{(t)}$ from u to v with cost t and probability $q_{uv}^{(t)}$ equal to the total probability of all length t walks from u to v all of whose states, except for the end points u and v, are in B. Since M is irreducible, any walk in M that enters B has to eventually leave it, and hence the new transitions define a Markov chain on V'. Call this new Markov chain M'. Since M is irreducible, so is M'.

Now, for any walk in M', define the cost of the walk to be the total cost of all transitions in the walk. The cost of a walk in M' is naturally mapped to the length of a corresponding walk taken in M (where taking one of the new transitions is to be interpreted as taking *all* the corresponding walks of length equal to the prescribed cost which are entirely in B, except for the end points).

This correspondence between walks in M and M' also implies that the stationary distribution in M' is one that assigns probability $\pi'_u = \pi_u/\pi(V')$ to state u. This can be seen by considering an arbitrary state $z \in V'$ and using the fact that the stationary probability of any state $u \in V'$ is proportional to the expected number of visits to u before returning to z, and then using the correspondence between the walks to argue that the expectation is the same in both M and M'.

Let l be an integer to be chosen later. We have the following lemma:

Lemma 1 *There exists a set $\tilde{B} \subseteq V'$ such that $\pi(\tilde{B}) \leq \pi(B)$, with the property that for all $v \in V' \setminus \tilde{B}$,*

$$\mathbb{E}[\text{cost of } l \text{ step walk in } M' \text{ from } v] \;\leq\; 2l.$$

PROOF: Fix any starting state $s \in V'$. Let $X^{(k)}$ be the k-th state on an l step walk in M' from s. For any $u \in V'$, let $p'^{(k)}_{su}$ be the probability of reaching u from s on step k of a random walk in M'.

For any $u \in V'$, we have

$$\mathbb{E}[\text{cost of step } k+1 \mid X^{(k)} = u] \;=\; \mathbb{E}[\text{cost of one step from } u],$$

by the Markov property. So define

$$c_u \;=\; \mathbb{E}[\text{cost of one step from } u] = \sum_{v \in V'} \left(p_{uv} + \textstyle\sum_{t \geq 2} q^{(t)}_{uv} \cdot t \right).$$

We have

$$\begin{aligned}
\mathbb{E}[\text{cost of } l \text{ step walk from } s] \;&\leq\; l + \sum_{k=1}^{l} \sum_{u \in V'} \Big[\mathbb{E}[\text{cost of step } k+1 \mid X^{(k)} = u] - 1 \Big] \cdot p'^{(k)}_{su} \\
&=\; l + \sum_{k=1}^{l} \sum_{u \in V'} (c_u - 1) \cdot p'^{(k)}_{su}.
\end{aligned}$$

Now, we define

$$\tilde{B} \;:=\; \left\{ s \in V' : \sum_{k=1}^{l} \sum_{u \in V'} (c_u - 1) \cdot p'^{(k)}_{su} \;\geq\; l \right\}. \tag{1}$$

With this definition, it is clear that for any starting state $s \in V' \setminus \tilde{B}$, we have

$$\mathbb{E}[\text{cost of } l \text{ step walk from } s] \;\leq\; 2l.$$

We proceed to bound $\pi(\tilde{B})$ as follows:

$$\begin{aligned}
l \cdot \pi(\tilde{B}) \;&\leq\; \sum_{s \in V'} \pi_s \sum_{k=1}^{l} \sum_{u \in V'} (c_u - 1) \cdot p'^{(k)}_{su} \\
&=\; \sum_{u \in V'} (c_u - 1) \cdot \sum_{k=1}^{l} \sum_{s \in V'} \pi_s p'^{(k)}_{su} \\
&=\; \sum_{u \in V'} (c_u - 1) \cdot \sum_{k=1}^{l} \pi_u \\
&=\; l \cdot \sum_{u \in V'} \pi_u (c_u - 1).
\end{aligned}$$

where the second-to-last equality follows because $\pi' = \pi' P'^k$, where P' is the transition matrix of M', which implies that $\sum_{s \in V'} \pi'_s p'^{(k)}_{su} = \pi'_u$. Note that $\pi'_u \propto \pi_u$ for $u \in V'$.

We now need to bound $\sum_{u \in V'} \pi_u (c_u - 1)$. Note that c_u is exactly the expected time to return to the set V' starting from u in the original Markov chain M. Since M is irreducible, by Kac's lemma (see [**?**]), we have

$$\sum_{u \in V'} \pi_u c_u \;=\; 1,$$

and hence

$$\sum_{u \in V'} \pi_u(c_u - 1) \;=\; 1 - \pi(V') \;=\; \pi(B).$$

Thus, we have $l \cdot \pi(\tilde{B}) \leq l \cdot \pi(B)$, which implies that $\pi(\tilde{B}) \leq \pi(B)$. □

Now, we would like to prove that the set $B' = \tilde{B} \cup B$, where $\tilde{B}$ is defined in Lemma 1 works for Theorem 2, when l is chosen to be the mixing time of M', which can be bound in terms of the conductance of M'. The idea is that even without the newly added transitions, the Markov chain V' has high conductance. Thus, if a short walk in V' mixes, then by Lemma 1, a corresponding short walk in the original chain should mix as well.

PROOF:[Theorem 2]
We use the notion of *stopping rules* for Markov chains [19]. A stopping rule is a rule that observes the walk and tells us whether to stop or not, depending on the walk so far (but independently of the continuation of the walk). This decision may be reached using coin flips, so the stopping rule has to only specify for each finite walk w, the probability of continuing the walk, so that with probability 1, the walk is stopped in a finite number of steps. The expected time for a stopping rule to terminate the walk yields bounds on the mixing time of the chain.

Let $B' = \tilde{B} \cup B$, where the set $\tilde{B}$ is as defined in equation (1) in Lemma 1. Let $s \in V \setminus B' = V' \setminus \tilde{B}$. We consider a random walk in the original Markov chain M starting from s with the following probabilistic stopping rule Γ: stop the walk as soon as it has taken l steps in the induced walk in M'. Note that this stopping rule always stops the walk on some state in V'. Denote by $\mathbb{E}_s[\Gamma]$ the expected number of steps the walk takes starting from s before being terminated by the stopping rule Γ.

Now, because the Markov chain M has V'-conductance at least ϕ, the Markov chain M' has conductance at least ϕ, and hence by the result of Sinclair [25], an $l = 2[\log(2/\epsilon\pi_0)]/\phi^2$ step walk starting from s mixes on V', i.e.

$$\|\vec{p'}_s^{(l)} - \pi'\|_{\mathrm{TV}} \;\leq\; \epsilon/2,$$

where $\vec{p'}_s^{(l)}$ is the probability vector for an l step random walk starting at s. Furthermore, we have $\|\pi' - \pi\|_{\mathrm{TV}} = \pi(B)$, and hence by the triangle inequality,

$$\|\vec{p'}_s^{(l)} - \pi\|_{\mathrm{TV}} \;\leq\; \epsilon/2 + \pi(B).$$

Furthermore, $\mathbb{E}_s[\Gamma] \leq 2l$ by Lemma 1. Thus, for all $s \in V \setminus B'$, the stopping rule Γ stops the walk in a distribution that is within total variation distance $\epsilon/2 + \pi(B)$ of the stationary distribution within an expected $2l$ steps. Lovász and Winkler [19] define $\mathcal{H}(s, d_\alpha)$ to be the minimal expected time for a stopping rule to stop a chain started from s in a distribution that is within variation distance α of the stationary distribution, and thus we have shown that

$$\mathcal{H}(s, d_{\epsilon/2+\pi(B)}) \;\leq\; 2l.$$

Now, Theorem 4.22 in [19] implies that for the uniform averaging chain of t steps in M started from s, if $\vec{p}_s$ is the final distribution, then

$$\|\vec{p}_s - \pi\|_{\mathrm{TV}} \;\leq\; \epsilon/2 + \pi(B) + \frac{1}{t}\mathcal{H}(s, d_{\epsilon/2+\pi(B)}) \;\leq\; \epsilon + \pi(B),$$

for $t \geq 4l/\epsilon$, which completes the proof. □

3 The Reconstruction Algorithm

We now give an overview of the reconstruction algorithm. There are two steps to the algorithm. In the first step, the algorithm tries to identify vertices that are part of a bad cut (i.e. one with low conductance), and in the second step, it attempts to boost the conductance of the cut by adding edges to the identified vertices. Care needs to be taken to selectively identify vertices only if they are present on the smaller side of a bad cut, otherwise we might end up adding too many edges. We use a random walk based procedure to separate vertices on the smaller side of a bad cut from the larger. The details of this separation procedure are given in Section 3.1.

Also, it doesn't suffice to identify any one particular bad cut, the edges need to be added carefully so that *any* bad cut is fixed. One way to do that is to add edges at random to the identified vertices. This works, but doesn't allow for parallel query processing, because the randomness needs to be fixed in advance for that. Instead, we use an explicit expander, and construct a hybrid of the original graph and the expander, which boosts the conductance of the final graph. The hybridization can be done locally, and this allows parallel query processing. Details of the hybridization procedure are given in Section 3.2.

3.1 Separating vertices

The first step to reconstruction is having a sublinear-time algorithm that classifies some vertices as *weak*. A weak vertex is (roughly speaking) one which is present in a bad cut and therefore needs edges to be added to it. In other words, the random walk from a weak vertex v has very bad mixing properties. Furthermore, we will define *strong* vertices as those from which a random walk has excellent mixing properties. Note that there can be vertices that are neither strong nor weak. The aim of our procedure will be to separate weak vertices from strong ones.

We now define a Markov chain on the graph, and we will study random walks in this Markov chain. The states of the chain are the nodes of the graph, and the edges represent transitions, where from each node, any outgoing edge is taken with probability $1/2d$. With the remaining probability, the walk stays at the current node. Note that this is an irreducible (assuming the graph is connected), aperiodic, and time-reversible chain, and the stationary distribution is uniform on all the nodes.

Let $\ell = c\log(n)/\phi^2$, where c is a large enough constant to be chosen later. Instead of considering random walks of length ℓ, we will consider the *uniform averaging walk* of length ℓ. We actually describe two kinds of random walks based on related probabilistic stopping rules:

1. Pick an integer $t \in \{0, 1, 2, \ldots, \ell - 1\}$ uniformly at random, and stop the walk after t steps. This is the uniform averaging walk of length ℓ. Let p_{uv} be the probability of reaching v from u after such a random walk.

2. Pick two integers $t_1, t_2 \in \{0, 1, 2, \ldots, \ell - 1\}$ uniformly at random, and stop the walk after $t_1 + t_2$ steps. Let q_{uv} be the probability of reaching v from u after such a random walk.

Let $\vec{p}_u$ be the probability vector of p_{uv}'s. For a subset of vertices S, let $p_u(S) := \sum_{v \in S} p_{uv}$ (similarly we define $\vec{q}_u$ and $q_u(S)$). By reversibility, it is easy to see that for any two vertices u and v, $q_{uv} = \sum_w p_{uw} \cdot p_{wv}$. For ease of notation, we will refer to the above walks as p-random walks and q-random walks.

In our procedures and definitions, we will use some constants. The constant c is some sufficiently large integer, and the constants $\alpha, \beta, \gamma, \delta, \sigma$ are sufficiently small positive numbers. We now give the definition of weak and strong vertices.

Definition 1 (Strong and Weak vertices)

1. *A vertex $u \in V$ is called* strong *if* $\|\vec{p}_u - \frac{\vec{1}}{n}\|_{TV} \leq \alpha$.
2. *A vertex $u \in V$ is called* weak *if* $\|\vec{q}_u - \frac{\vec{1}}{n}\|_{TV} \geq 1/4$.

Intuitively, strong vertices are those that reach a vast majority of vertices (through short p-walks) with probability $\Omega(1/n)$. To distinguish between strong and weak nodes, we need to have a procedure that can, in sublinear time, estimate the q_{uv} probabilities. The basic idea is to perform many p-walks from u (and v) and compute the number of collisions between these walks (at the endpoints): this is a twist on the idea of Goldreich and Ron [16] for using random walks to estimate the mixing of a random walk. A birthday paradox like argument suggests that $O(\sqrt{n})$ walks should be sufficient to estimate probabilities of $\Omega(1/n)$.

Unfortunately, assuming nothing about the vectors $\vec{p}_u$ and $\vec{p}_v$, we cannot get a reasonable bound on the variance of our randomized estimate[3]. For this reason, we define the *reduced collision probability*, which disregards collisions which occur due to vertices which are reached with extraordinarily high probability. This quantity attempts to approximate q_{uv} and can be estimated quite accurately in sublinear time. We will show that for the purpose of separating weak vertices from strong ones, it suffices to estimate these probabilities. This sublinear time algorithmic technique for estimating probabilities of going from one vertex to another should be applicable in other scenarios too.

Definition 2 (Reduced Collision Probability) *Consider vertices u, v. Let $S_u^\sigma = \{w : p_{uw} \leq (1-\sigma)/\sqrt{n}\}$ (set S_v^σ is similarly defined). The* σ-reduced collision probability *of vertices u, v (denoted by $\mathtt{rcp}_{uv}^\sigma$) is:*

$$\mathtt{rcp}_{uv}^\sigma = \sum_{w \in S_u^\sigma \cap S_v^\sigma} p_{uw} p_{vw}$$

Lemma 2 *The following properties of strong vertices hold:*

1. *For any strong vertex v, there can be at most $\sqrt{\alpha}n$ vertices with $p_{uv} \leq (1-\sqrt{\alpha})/n$.*
2. *For any strong vertex u and any $\sigma \leq \frac{1}{2}$, we have $p_u(S_u^\sigma) \geq 1/2$.*
3. *Let u and v be strong vertices. Then for any $\sigma \leq \frac{1}{2}$, we have $\mathtt{rcp}_{uv}^\sigma \geq (1-\beta)/n$ for $\beta \geq 3\sqrt{\alpha}$.*

Proof: The first part follows immediately from the definition of strong vertices. For the second part, note that there can be at most $2\sqrt{n}$ vertices in $V \setminus S_u^\sigma$. Thus, there are at least $(1-\sqrt{\alpha})n - 2\sqrt{n}$ vertices $v \in S_u^\sigma$ such that $p_{uv} \geq (1-\sqrt{\alpha})/n$, and hence

$$p_u(S_u^\sigma) \;\geq\; [(1-\sqrt{\alpha})n - 2\sqrt{n}] \cdot (1-\sqrt{\alpha})/n \;\geq\; 1/2.$$

We now prove the third part. There are at least $(1-\sqrt{\alpha})n - 2\sqrt{n}$ vertices $w \in S_u^\sigma$ such that $p_{uw} \geq (1-\sqrt{\alpha})/n$, and a similar statement holds for v. Thus, there are at least $(1-2\sqrt{\alpha})n - 4\sqrt{n}$ vertices in $S_u^\sigma \cap S_v^\sigma$ such that both p_{uw} and p_{vw} are at least $(1-\sqrt{\alpha})/n$. Hence,

$$\mathtt{rcp}_{uv}^\sigma \;\geq\; [(1-2\alpha)n - 4\sqrt{n}] \cdot [(1-\sqrt{\alpha})/n]^2 \;\geq\; (1-3\sqrt{\alpha})/n.$$

□

[3]Think of the extreme situation where for some w, p_{uw} is a constant, whereas all coordinates of $\vec{p}_v$ are $\Theta(1/n)$. Although $q_{uv} = \Omega(1/n)$, it is a very unlikely that a sublinear time procedure can detect a collision between p-walks from u and v.

We now describe a procedure ESTIMATE-RCP that outputs an estimate of $\texttt{rcp}_{uv}$. The procedure ESTIMATE-RCP needs another procedure, FIND-SET, which given a vertex u, finds the set S_u^σ (approximately). Note that what we actually require is just an oracle that given a vertex v, tells us whether or not it is in S_u^σ. Since S_u^σ is of linear size, the procedure actually outputs the complement (which is of size $O(\sqrt{n})$.

FIND-SET
Input: Vertex $u \in V$.
Parameters: c, σ, ℓ.

1. Perform $c\sqrt{n}\log n$ independent p-random walks of length ℓ from u.
2. Return $\hat{S}_u$, the set of all vertices w such that at most $c(1-\sigma/2)\log n$ p-random walks from u end at w.

ESTIMATE-RCP
Input: Vertices $u, v \in V$.
Parameters: δ, $m = \sqrt{n}/\delta^2$.

1. Let $\hat{S}_u$:=FIND-SET(u), and $\hat{S}_v$:=FIND-SET(v).
2. Keep performing p-random walks from u until m such walks end at vertices in $\hat{S}_u$ (call this set of walks W_u). If more than $20m$ walks are performed, then ABORT.

   Do the same for walks starting from v as well to obtain the set of m walks W_v.
3. Let A be the number of pairwise collisions between walks in W_u and W_v (if a walk from W_u and a walk from W_v end at the same vertex, then this counts as a pairwise collision). Output A/m^2.

Lemma 3 *With probability at least $1-1/n$, the procedure* FIND-SET*, on input vertex u, outputs a set $\hat{S}_u$ such that*

$$S_u^\sigma \;\subseteq\; \hat{S}_u \;\subseteq\; S_u^0.$$

PROOF: Any vertex $w \in S_u^\sigma$ has $p_{uw} \le (1-\sigma)/\sqrt{n}$, and any vertex $v \in V \setminus S_u^0$ has $p_{uv} > 1/\sqrt{n}$. Since we run $c\sqrt{n}\log n$ random walks from u, the expected number of times such a walk hits w is at most $c(1-\sigma)\log n$, and the expected number of times such a walk hits v is at least $c\log n$. Thus, by a Chernoff bound, the chance we have more than $c(1-\sigma/2)\log n$ walks hitting w and less than $c(1-\sigma/2)\log n$ walks hitting v is at most $1/n^2$, by making c large enough. The statement of the lemma follows by a union bound. □

Lemma 4 *Let u and v be two vertices. The running time of* ESTIMATE-RCP *is $O(\sqrt{n}\ell\log n)$. If both $p_u(S_u^\sigma), p_v(S_v^\sigma) \ge 1/2$, then* ESTIMATE-RCP *aborts with probability less than $\exp(-O(m))$. If* ESTIMATE-RCP *does not abort, then it outputs an estimate $\tilde{\texttt{rcp}}$ such that with probability at least 9/10,*

$$\texttt{rcp}_{uv}^\sigma - \delta\max\{\texttt{rcp}_{uv}^\sigma, 1/2n\} \;\le\; \tilde{\texttt{rcp}} \;\le\; \texttt{rcp}_{uv}^0 + \delta\max\{\texttt{rcp}_{uv}^0, 1/2n\}.$$

PROOF: The running time bound is obvious, since the algorithm runs $O(\sqrt{n}\log n)$ random walks from u and v. By Lemma 3, with high probability, we have

$$S_u^\sigma \subseteq \hat{S}_u \subseteq S_u^0 \quad \text{and} \quad S_v^\sigma \subseteq \hat{S}_v \subseteq S_v^0.$$

Thus, if $p_u(S_u^\sigma) \geq 1/2$, by a Chernoff bound, the probability that ESTIMATE-RCP aborts while performing walks from u is less than $\exp(-O(m))$. This proves the first part.

Define $\hat{\texttt{rcp}}_{uv}$ as:

$$\hat{\texttt{rcp}}_{uv} = \sum_{w \in \hat{S}_u \cap \hat{S}_v} p_{uw} p_{vw}$$

Since (with high probability), $S_u^\sigma \cap S_v^\sigma \subseteq \hat{S}_u \cap \hat{S}_v \subseteq S_u^0 \cap S_v^0$, we get that $\texttt{rcp}_{uv}^\sigma \leq \hat{\texttt{rcp}}_{uv} \leq \texttt{rcp}_{uv}^0$. Assuming that the algorithm does not ABORT, let X be the random variable denoting number of pairs of walks that collide. The algorithm's estimate is X/M, where $M = m^2$. We now show that this is a good estimate, with a high probability. For any pair of walks, the probability that they collide is $\sum_{w \in \hat{S}_u \cap \hat{S}_v} p_{uw} \cdot p_{vw} = \hat{\texttt{rcp}}_{uv}$. Thus, the expectation of X is exactly $\hat{\texttt{rcp}}_{uv} M$. We can show that $\mathbf{Var}(X) \leq 4\hat{\texttt{rcp}}_{uv} M^{3/2}/\sqrt{n}$ (see Lemma 8 in the appendix). Thus, by Chebyshev's inequality, if $q = \max\{\hat{\texttt{rcp}}_{uv}, 1/2n\}$, then

$$\mathbf{Pr}[|X - \hat{\texttt{rcp}}_{uv} M| > (\delta q)M] \;\leq\; \frac{4\hat{\texttt{rcp}}_{uv} M^{3/2}}{\sqrt{n}[(\delta q)M]^2} \;\leq\; 1/10$$

since $M = n/\delta^4$, and we can choose δ to be a small enough constant. This implies that with probability at least 9/10,

$$\texttt{rcp}_{uv}^\sigma - \delta \max\{\texttt{rcp}_{uv}^\sigma, 1/2n\} \;\leq\; \tilde{\texttt{rcp}} \;\leq\; \texttt{rcp}_{uv}^0 + \delta \max\{\texttt{rcp}_{uv}^0, 1/2n\}.$$

□

Using the procedure ESTIMATE-RCP, we can now design a procedure SEPARATE that distinguishes strong vertices from weak ones. The basic idea is that a strong vertex reaches many vertices in ℓ-length random walks with probability close the $1/n$, whereas a weak vertex does not have this property. Although, we cannot directly estimate the probability of going from one vertex to another in sublinear time, the $\texttt{rcp}$ value gives us (in most circumstances) a close enough approximation.

SEPARATE
Input: Vertex $u \in V$.
Parameters: β, γ, δ, ℓ

1. Choose a random set R of $c \log n/\gamma$ vertices.
2. For every $v \in R$, run ESTIMATE-RCP(u, v), $c \log n$ times. If for a majority of these runs, ESTIMATE-RCP does not abort and outputs $\hat{\texttt{rcp}} \geq (1-\beta)(1-\delta)/n$, then call v *accessible*.
3. If more than a $(1-2\gamma)$-fraction of vertices in R are accessible, then ACCEPT. Otherwise REJECT.

Lemma 5 *The procedure Separate runs in $O(\sqrt{n}\ell \log^2 n)$ time, and with probability $> 1 - n^{-3}$ has the following behavior:*

1. *Assume there are at least $(1-\gamma)n$ strong vertices, for some constant γ. If u is strong, then the algorithm accepts u.*
2. *If u is weak, then the algorithm rejects u.*

PROOF: For the first part, we assume that there are at least $(1-\gamma)n$ strong vertices. We now show that if u is a strong vertex, then this test will accept it with probability of error less than $1/n^3$. Since the sample size is $c \log n/\gamma$ (for sufficiently large c), a Chernoff bound argument shows that with probability of error less than $1/n^4$, at least a $(1-2\gamma)$ fraction of vertices in the sample must be strong.

Let v be a strong vertex in the sample. By Lemma 2, parts 2 and 3, we have $p_u(S_u^\sigma) \geq 1/2$, $p_v(S_v^\sigma) \geq 1/2$ and $\texttt{rcp}_{uv}^\sigma \geq (1-\beta)/n$. With probability at least 2/3, ESTIMATE-RCP will not abort and will output a value that is at least $(1-\beta)(1-\delta)/n$. Hence, with probability of error less than $1/n^4$, v will be deemed accessible. Taking a union bound over all errors, u is accepted with probability at least $1-1/n^3$.

Now we turn to the second part. Let u be a weak vertex. Then it is easy to see from the definition of weak vertices that there can be at most $7n/8$ vertices v with $q_{uv} \geq 7/8n$. Suppose this is not the case. Let S be the set all the vertices v with $q_{uv} < 1/n$. At least $7n/8$ have $q_{uv} \geq 7/8n$. Thus,

$$\|q - \vec{1}/n\|_{\text{TV}} \;\leq\; \sum_{v \in S}(1/n - q_{uv}) \leq (7n/8)\cdot(1/8n) + (n/8)\cdot(1/n) \;<\; 1/4,$$

a contradiction. By the relation between p and q probabilities, $\texttt{rcp}_{uv}^0 \leq q_{uv}$, so again, there can be at most $7n/8$ vertices v with $\texttt{rcp}_{uv}^0 \geq 7/8n$. Now, in our random sample, (with probability of error less than $1/n^4$) a 1/9 fraction of vertices v must have $\texttt{rcp}_{uv}^0 \leq 7/8n$. Consider any such v. If ESTIMATE-RCP does not abort, then with probability at least 2/3, it will output an estimate of $\texttt{rcp}_{uv}^0$ that is at most $7/8n + \delta/2n$. By choosing constants α, β, δ to be sufficiently small, we can ensure that $7/8n + \delta/2n < (1-\beta)(1-\delta)/n$. Thus, with probability of error less than $1/n^4$, the algorithm will not call v accessible. A union bound over all v implies that with probability at least $1-1/n^3$, a 1/9 fraction of vertices in the random sample are not accessible from u, and if we choose $2\gamma < 1/9$, then the algorithm rejects u. □

Henceforth, we will just assume that all strong vertices are accepted and all weak vertices are rejected.

3.2 Hybridizing the Graph with an Expander

Now that we have a separating procedure, we can describe the actual reconstruction procedure which hybridizes an expander with our original graph. Given a query vertex v, the procedure will output at most d vertices which will be the neighbors of v in the reconstructed graph. We will refer to the final reconstructed graph as G', and for cuts $(S, \bar{S})$ in G', we use the notation $E'(S, \bar{S})$ to refer to the set of edges crossing the cut.

Since we are describing a parallel filter, we assume we have access to a sublinear sized random seed s of size $\widetilde{O}(\sqrt{n})$ which is fixed for all queries. Since the total number of calls to SEPARATE is at most $O(n)$ (for each call to the reconstruction procedure, we make $O(1)$ calls to SEPARATE), by taking a union bound over all the error probabilities, we can ensure that the guarantees of Lemma 5 hold with probability at least $1-1/n^2$. Since the seed is fixed, we can unambiguously refer to vertices as *accepted* or *rejected*, based on SEPARATE.

For constructing G', we use an explicit bounded degree expander G^* with n vertices. The explicit construction allows us to find all neighbors of a vertex $v \in G^*$ in $\text{poly}(\log n)$ time. Naturally, there is a one-to-one correspondence between the vertices of G and G^*. Abusing notation, given a vertex v in G, we will refer to the corresponding vertex in G^* also as v. To prevent confusion about edges, we will call edges G, G^*, or G'-edges depending on the graph in consideration.

For the sake of intuition, we can think of the accepted vertices as strong, and the rejected as weak. The reconstructed graph G' will be a careful combination of G and G^*. Starting with G,

here is an informal description of how G' is built. For any rejected vertex v, we remove all G-edges incident to v and add all G^*-edges incident to v to get the G^*-edges. This is to ensure that any subset of rejected vertices will have large conductance. For accepted vertices, we would like to just keep the same G-edges.

This construction could potentially make some (accepted) vertices have a degree larger than d. Therefore, we have to remove some G-edges incident to accepted vertices to maintain the degree bound. These edges are removed based on a simple *local* rule. Note that this choice cannot be arbitrary, since we want a parallel filter (for example, if the G-edge (u, v) is removed on querying v, then it must also be removed on querying u). Unfortunately, this might affect the condutance of subsets of accepted vertices. We ensure that every time we remove such an edge, we replace it by a very short path (again decided by local considerations) between the endpoints without affecting the degree bound. This gives us G' with the desired properties. Because we want every query to be handled in sublinear time, we give a procedure that determines the neighbors of a vertex in G' by running SEPARATE on a constant number of vertices.

We now give a formal description of the hybridization process. We assume that the explicit expander G^* is a strong edge expander with the following property:

Property 1 *The expander G^* has degree bound $d/2$. For any set of vertices S in G^* with $|S| \leq n/2$, we have $|E^*(S, \bar{S})| \geq \eta|S|d$, where η is a constant.*

We assume that there is some global ordering of the vertices (say, according to the value of their indices). Thus, given any vertex v, there is an *ordered* list of the neighbors of v, with possibly some "null" entries at the end (because v might have degree less than d). When we refer to the i^{th} vertex in some list of vertices, we mean the i^{th} vertex in the list in the global ordering.

As mentioned before, if any vertex u is rejected by SEPARATE, we remove all the G-edges incident on it and replace them by G^*-edges. To avoid increasing the degree of accepted vertices, we need to remove some G-edges between accepted vertices as well. We now describe a procedure that given an edge $e = (u, v)$ of G, where u and v are both accepted by SEPARATE, outputs whether the edge needs to be removed, and if so, what edges are added (because of this removal). The procedure REMOVE involves $O(d)$ calls to SEPARATE.

REMOVE
Input: Edge $(u, v) \in G$.

1. Find the G^*-neighbors of u and v and consider all the rejected vertices (according to SEPARATE).
2. Suppose v is the i^{th} neighbor of u, and u is the j^{th} neighbor of v. Let s be i^{th} rejected G^*-neighbor of u, and let t be the j^{th} rejected G^*-neighbor of v. Note that one or both of s and t could be null.
3. If both s and t are null, then (u, v) is not removed. If t is null, then the edges (s, u) and (s, v) replace (u, v). If s is null, then the edges (t, u) and (t, v) replace (u, v). If neither s nor t is null, then the edges $(u, s), (s, t), (t, v)$ replace (u, v). In any case, output the new neighbors to u and v.

Note that if an edge (u, v) is replaced, it is replaced by a path from u to v. Furthermore, for two different edges, these paths are edge-disjoint. This gives the following claim.

Claim 1 $|E'(S, \bar{S})| \geq |E(S, \bar{S})|$

Using the procedure REMOVE, we can describe the main reconstruction procedure. This procedure will output all the neighbors of an input vertex u on the reconstructed graph and involves calling REMOVE on $O(d)$ edges.

RECONSTRUCT
Input: Vertex $v \in V$.

1. Using SEPARATE, check if v is accepted or rejected.
2. If v is accepted : Call REMOVE on all the G-edges incident on v. All these outputs give the neighbors of v in the reconstructed graph.
3. If v is rejected : Remove all G-edges incident on v. Find the G^*-neighbors of v. For each such u:
   (a) If u is rejected, then add edge (u, v) to G'.
   (b) If not, suppose v is the i^{th} rejected G^*-neighbors of u. Let w be the i^{th} G-neighbor of u. If w is rejected by SEPARATE, then add (u, v). Otherwise, call REMOVE on (u, w).

   All these calls together will output the neighbors of v.

Claim 2 *The procedure* RECONSTRUCT *is consistent: vertex v is output as a neighbor of u (in G') iff u is output as a neighbor of v.*

PROOF: Suppose u is an accepted vertex and let v be output as a G'-neighbor of u. The neighbor v came about because of a call to REMOVE on a G-edge (u, w). If (u, w) is not removed, then w and v are the same. In this case, u will also be output as a neighbor of v in G'. Suppose (u, w) is replaced by (u, v) and (w, v) for some rejected vertex v. Then, for some i, v is the i^{th} rejected G^*-neighbor of u (or w), and w is the i^{th} G-neighbor of u (or vice versa). When RECONSTRUCT is called on v, then we can see that REMOVE will be called on (u, w), and u will be output as a G'-neighbor of v.

Now suppose that (u, w) is replaced by $(u, s), (s, t), (t, w)$, where s, t are rejected vertices. Using a similar argument as the one above, we can show that u will be output as a G'-neighbor of v.

Let u be a rejected vertex. If v is a rejected vertex that is a G'-neighbor of u, then u will also be output as a G'-neighbor of v. If v is accepted, then we can make the above argument to prove consistency. □

Lemma 6 *The procedure* RECONSTRUCT *and the final graph G' have the following properties:*

1. RECONSTRUCT *is consistent over parallel queries and outputs the graph G'.*
2. *The running time of* RECONSTRUCT *for any query vertex is $O(\sqrt{n}\ell \log^2 n)$ and a random seed s of at most $O(\sqrt{n}\ell \log^2 n)$ bits is required.*
3. *The graph G' has degree bound d.*
4. *The conductance of any set S in G' is larger than the conductance of S in G.*
5. *The set of G'-neighbors of a weak vertex u is a superset of the set of G^*-neighbors of u.*

4 Bounding the number of edges added

In this section, we show the connection between the number of weak vertices to the optimal number of edges to be changed to make the conductance of G at least ϕ. One can look at this results as drawing a connection between two different notions of measuring the "distance" to having a large conductance. The obvious measure is the amount of change in terms of number of edges that needs to be done to boost the conductance. We show that the number of weak vertices, which can also been seen as a measure of how far G is from having a large condutance, is comparable to this distance.

First, we show that there is a large cut of low conductance:

Lemma 7 *Let S be the set of strong vertices. Then there is a cut $(B, V\backslash B)$ with the property that $\Omega(n - |S|) \leq \min\{|B|, |V \setminus B|\}$, which has conductance less than $\phi/2$.*

Proof: We keep recursively partitioning the graph, finding cuts in the remaining induced graph of conductance less than $\phi/2$, and aggregating these cuts. Our goal is to show that the final cut obtained has the properties required here.

To be more precise: start out with $B = \{\}$. Let $\bar{B} = V \setminus B$. If there is a cut $(S, \bar{S})$ in $\bar{B}$ with $|S| \leq |\bar{B}|/2$ having conductance less than $\phi/2$, then we set $B := B \cup S$, and continue, as long as $|B| \leq n/2$. If $|B|$ exceeds $\frac{\alpha}{2}n$, then note that $|B| \leq (\frac{1}{2} + \frac{\alpha}{4})n$, and we are done since $\min\{|B|, |V \setminus B|\} \geq \Omega(n) \geq \Omega(n - |S|)$.

We therefore assume that $|B| \leq \frac{\alpha}{2}n$. It is easy to check that the final cut $(B, \bar{B})$ also has conductance less than $\phi/2$. Furthermore, the subgraph induced on $\bar{B}$ has conductance at least $\phi/2$, or in other words, the $\bar{B}$-conductance of G is at least $\phi/2$.

Now, we apply Theorem 2 to G, to conclude that there is a set B' such that $|B'| \leq 2|B|$ (because the stationary distribution is uniform) with the property that starting from any $s \in V \setminus B'$, the uniform averaging walk, after ℓ steps, ends up in a distribution $p_s^{(\ell)}$ such that

$$\|p_s^{(\ell)} - \vec{1}/n\|_{\text{TV}} \;\leq\; \alpha/2 + |B|/n, \tag{2}$$

since $\ell \geq 64\log(4n/\alpha)/(\alpha\phi^2)$. Since $|B| \leq \frac{\alpha}{2}n$, then inequality (2) implies that $\|p_s^{(\ell)} - \vec{1}/n\|_{\text{TV}} \leq \alpha$ for all $s \in V \setminus B'$, which implies that all such vertices s are strong. Hence, $|S| \geq n - |B'| \geq n - 2|B|$, and so $\min\{|B|, |V \setminus B|\} = |B| \geq \Omega(n - |S|)$ in this case as well. □

This lemma enables us to bound the number of edges added by the reconstruction algorithm in terms of the optimal reconstruction:

Theorem 3 *The reconstruction algorithm achieves an approximation ratio of $O(1/\phi)$ to the optimal number of edges to be changed to make the conductance of the graph at least ϕ.*

Proof: Lemma 7 immediately implies that the optimal reconstruction of the graph must add at least $\Omega(\phi d(n - |S|))$ edges to patch up the cut $(B, V \setminus B)$. Whenever the reconstruction algorithm adds an edge, then one of the endpoints is a rejected vertex. The total number of removed edges is at most the number of added edges. Therefore, we can bound the total change (up to constant factors) by d times the number of rejected vertices. Suppose the number of strong vertices is less than $(1 - \gamma)n$. The graph G is trivially changed by atmost $O(nd) = O(d(n - |S|))$ edges. If the number of strong vertices is more than $(1-\gamma)n$, then by Lemma 5 all strong vertices are accepted. Our reconstruction algorithm potentially adds $O(d(n - |S|))$ edges, which means that we have an approximation ratio of $O(1/\phi)$ to the optimal number of edges to be changed. □

5 Conductance bound of the reconstructed expander

Theorem 4 *The reconstructed graph G' has conductance at least $\Omega(1/\ell)$.*

PROOF: Consider a cut $(S, \bar{S})$ in the reconstructed graph G', with $|S| \leq n/2$. Suppose S has a subset T of more than $(1-\eta/2)|S|$ weak vertices (where η comes from Property 1). By the properties in Lemma 6, all the edges incident to these weak vertices in G^* are present in G'. The number of edges leaving T is at least $\eta|T|d$, whereas the number of edges incident on $S \backslash T$ is at most $\eta|T|d/2$. This implies that the conductance of S is $\Omega(1)$.

So assume that less than a $(1-\eta/2)$-fraction of the vertices in S are weak. Then we claim that the conductance of S is already at least $\eta/16\ell$. By Lemma 6, it suffices to prove this for G. Assume for the sake of contradiction that the conductance of S (in G) is less than $\eta/16\ell$. Then the following modification of the proof of Lemma 4.7 in Czumaj-Sohler [11] gives us the desired contradiction.

Consider a q-random walk starting from the uniform distribution on the vertices. Since the uniform distribution is stationary, the i^{th} vertex is uniformly distributed in V. Thus, for any edge that is not a self-loop, the chance that it is selected in the next step of the random walk is $(1/n) \cdot (1/2d) \cdot 2 = 1/nd$. Thus, we have

$$\mathbf{Pr}[\text{an edge in } (S, \bar{S}) \text{ is used in the step } i] \;\leq\; \frac{E(S, \bar{S})}{nd}.$$

Since the walk runs for at most 2ℓ steps, the chance of using an edge in the cut is at most $\frac{2\ell E(S,\bar{S})}{nd}$. The chance that the walk starts in S is $\frac{|S|}{n}$. Since the conductance of the cut $E(S, \bar{S})/d|S| < \eta/16\ell$, we get that

$$\mathbf{Pr}[\text{walk starts in } S \text{ and doesn't use an edge in the cut } (S, \bar{S})] \;\geq\; (1-\eta/8)\frac{|S|}{n}.$$

Thus, conditioned on starting in S, the chance that the walk never crosses the cut (or in other words, never leaves the set S) is at least $(1-\eta/8)$.

Therefore, for at least a $(1-\eta/2)$-fraction of vertices in S, the chance that the random walk starting from them never leaves the set S is at least $3/4$. On the set $\bar{S}$, the uniform distribution places a mass of at least $1/2$ (since $|\bar{S}| \geq 1/2$), while the q-random walk from one of the aforementioned vertices $u \in S$ places a mass of at most $1/4$, which implies that $\|\vec{q}_u - \frac{\vec{1}}{n}\|_{\text{TV}} \geq 1/4$. This implies that u is weak, and thus S contains at least a $(1-\eta/2)$-fraction of weak vertices, a contradiction. □

6 Further directions

One of the main direction of future research is improvement of the conductance bound of G' that we guarantee. It would be very interesting to design a filter than can output a graph of conductance ϕ^2 (instead of $\phi^2/\log n$). Such a result is not possible with the current definitions of weak and strong, and would require much more tighter characterizations. To get a final conductance of ϕ^2, we need to have definitions of weak/strong that distinguish vertices on the basis of very small probability differences (much smaller than we currently do). These differences can be algorithmically tested, but to ensure that not too many edges are added to get G', we would need much stronger results about walks in noisy expanders: again our statements about mixing must deal with probabilities

differences and discrepancy norms which are much smaller. Such a result would be interesting in its own right, and would probably use a host of new techniques and tools for studying noisy expanders.

It seems highly likely that the algorithmic procedures we use here could be used for efficient graph partitioning algorithms [4, 5, 6]. The partitions would be decided by performing random walks from vertices, and might lead to easier proofs of earlier results. We also may be able to generate sublinear routines which can implicitly represent such a partition: answering queries such as whether two vertices are in the same graph of the partition or not. Improved conductance bounds for reconstruction may lead to better graph partitioning algorithms.

References

[1] N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Property preserving data reconstruction. *Proc. of 15th ISAAC*, pages 16–27, 2004.

[2] N. Alon, E. Fischer, I. Newman, and A. Shapira. A combinatorial characterization of the testable graph properties : it's all about regularity. *Proc. 38th STOC*, pages 251–260, 2006.

[3] N. Alon and A. Shapira. A charaterization of the (natural) graph properties testable with one-sided error. *Proc. 46th FOCS*, pages 429–438, 2005.

[4] R. Andersen. A local algorithm for finding dense subgraphs. *Proc. of 19th SODA*, pages 1003–1009, 2008.

[5] R. Andersen, F. R. K. Chung, and K. Lang. Local graph partitioning using pagerank vectors. *Proc. of 47th FOCS*, pages 475–486, 2006.

[6] R. Andersen and K. Lang. An algorithm for improving graph partitions. *Proc. of 19th SODA*, pages 651–660, 2008.

[7] T. Batu, L. Fortnow, R. Rubinfeld, W. D. Smith, and P. White. Testing that distributions are close. *Proc. of 41st FOCS*, pages 259–269, 2000.

[8] I. Benjamini, O. Schramm, and A. Shapira. Every minor-closed property of sparse graphs is testable. *arxiv*, arXiv:0801.2797v2, 2008.

[9] B. Chazelle and C. Seshadhri. Online geometric reconstruction. *Proc. of 22nd SOCG*, pages 386–394, 2006.

[10] A. Czumaj and C. Sohler. On testable properties in bounded degree graphs. *Proc. 18th SODA*, pages 494–501, 2007.

[11] A. Czumaj and C. Sohler. Testing expansion in bounded degree graphs. *Proc. 48th FOCS*, pages 570–578, 2007.

[12] E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of EATCS*, 75:97–126, 2001.

[13] O. Goldreich. Combinatorial property testing - a survey. *Randomization Methods in Algorithm Design*, 75:45–60, 1998.

[14] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

[15] O. Goldreich and D. Ron. A sublinear bipartite tester for bounded degree graphs. *Combinatorica*, 19(3):335–373, 1999.

[16] O. Goldreich and D. Ron. On testing expansion in bounded-degree graphs. *ECCC*, TR00-020, 2000.

[17] O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002.

[18] S. Kale and C. Seshadhri. Testing expansion in bounded degree graphs. In *Proc. of the 35th ICALP*, pages 527–538, 2008.

[19] L. Lovász and P. Winkler. Mixing times. *Microsurveys in Discrete Probability, DIMACS Series in Discrete Math. and Theor. Comp. Sci., AMS*, pages 85–133, 1998.

[20] A. Nachmias and A. Shapira. Testing the expansion of a graph. *ECCC*, TR07-118, 2007.

[21] M. Parnas, D. Ron, and R. Rubinfeld. Tolerant property testing and distance approximation. *ECCC*, TR04-010, 2004.

[22] D. Ron. Property testing. *Handbook on Randomization*, II:597–649, 2001.

[23] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

[24] M. Saks and C. Seshadhri. Parallel monotonicity reconstruction. *Proc. of 19th SODA*, pages 962–971, 2008. Full version is titled "Distributed monotonicity reconstruction".

[25] A. Sinclair. Improved bounds for mixing rates of markov chains and multicommodity flow. *Combinatorics, Probability & Computing*, 1:351–370, 1992.

A Variance bound

Lemma 8 *The variance of X is bounded by* $\mathbf{Var}(X) \leq 4\hat{\mathtt{rcp}}_{uv} M^{3/2}/\sqrt{n}$.

Proof: For convenience of notation, let $p := \hat{\mathtt{rcp}}_{uv}$. We index the walks from u by $i = 1, 2, \ldots, m$, and the walks from v by $j = 1, 2, \ldots, m$. Define indicator random variables X_{ij} for any $i, j \in \{1, 2, \ldots, m\}$ which are set to 1 if the i^{th} walk from u and the j^{th} walk from v collide. Note that $\mathbf{Pr}[X_{ij} = 1] = p$. The random variable X equals $\sum_{ij} X_{ij}$, and thus the expectation of X is $pm^2 = pM$. We now estimate the variance of X as follows:

$$\mathbb{E}[X^2] = \sum_{ij} E[X_{ij}^2] + 2 \sum_{i \neq i', j \neq j'} E[X_{ij} X_{i'j'}] + 2 \sum_{i, j \neq j'} E[X_{ij} X_{ij'}] + 2 \sum_{j, i \neq i'} E[X_{ij} X_{i'j}].$$

We estimate each term separately. First, $E[X_{ij}^2] = p$. Then, if $i \neq i'$ and $j \neq j'$, then X_{ij} and $X_{i'j'}$ are independent, and hence $E[X_{ij} X_{i'j'}] = p^2$.

The last case is if $i = i'$ and $j \neq j'$ (the case when $i \neq i'$, $j = j'$ is handled analogously). Let p_{uw} (resp., p_{vw}) be the probability that a walk from u ends up at w (resp., probability that a walk

from v ends up at w). Then, we have

$$\begin{aligned} E[X_{ij}X_{ij'}] &= \mathbf{Pr}[X_{ij}X_{ij'} = 1] \\ &= \sum_{w \in \hat{S}_u \cap \hat{S}_v} p_{uw} p_{vw}^2 \\ &\leq \sum_{w \in \hat{S}_u \cap \hat{S}_v} \frac{1}{\sqrt{n}} \cdot p_{uw} p_{vw} \\ &= \frac{1}{\sqrt{n}} \cdot p. \end{aligned}$$

Thus, we have

$$\begin{aligned} \mathbf{Var}[X] &= E[X^2] - E[X]^2 \\ &\leq m^2 p + m^2(m-1)^2 p^2 + 2m^2(m-1) \cdot \frac{p}{\sqrt{n}} - m^4 p^2 \\ &\leq 4m^3 \cdot \frac{p}{\sqrt{n}} && \because m \gg \sqrt{n} \\ &= 4pM^{3/2}/\sqrt{n}. \end{aligned}$$

□