Bounded-Independence Sampling of Edges for Combinatorial Graph Properties

Aaron Putterman
Harvard University, Cambridge, MA, USA

Salil Vadhan
Harvard University, Cambridge, MA, USA

Vadim Zaripov
Harvard University, Cambridge, MA, USA

Abstract

Random subsampling of edges is a commonly employed technique in graph algorithms, underlying a vast array of modern algorithmic breakthroughs. Unfortunately, using this technique often leads to randomized algorithms with no clear path to derandomization because the analyses rely on a union bound over exponentially many events. In this work, we revisit this goal of *derandomizing* randomized sampling in graphs.

We give several results related to bounded-independence edge subsampling, and in the process of doing so, generalize several of the results of Alon and Nussboim (FOCS 2008), who studied bounded-independence analogues of random graphs (which can be viewed as edge subsamples of the complete graph). Most notably, we show:

1. $O(\log(m))$-wise independence suffices for preserving connectivity when sampling at rate $1/2$ in a graph with minimum cut $\geq \kappa \log(m)$ with probability $1 - 1/\text{poly}(m)$ (for a sufficiently large constant κ).
2. $O(\log(m))$-wise $(1/\text{poly}(m))$-almost independence suffices for ensuring cycle-freeness when sampling at rate $1/2$ in a graph with minimum cycle length $\geq \kappa \log(m)$ with probability $1-1/\text{poly}(m)$ (for a sufficiently large constant κ).
3. If we relax to *arbitrary* distributions, we show there is an explicit distribution with marginals $\leq 1/2$ generated using $O(\log(m) \log\log(m))$ random bits such that in a graph with minimum cut $\geq \kappa \log(m)$ (for a sufficiently large constant κ), a sample from the distribution has is still connected with probability $1 - 1/\text{poly}(m)$.

To demonstrate the utility of our results, we revisit the classic problem of using parallel algorithms to find graphic matroid bases, first studied in the work of Karp, Upfal, and Wigderson (FOCS 1985). In this regime, we show that the optimal algorithms of Khanna, Putterman, and Song (arxiv 2025) can be *explicitly* derandomized while maintaining near-optimality.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Graphs, random sampling

Digital Object Identifier 10.4230/LIPIcs.CCC.2026.2

Related Version *Full Version*: https://arxiv.org/abs/2603.25095

Funding *Aaron Putterman*: Supported in part by the Simons Investigator awards of Madhu Sudan and Salil Vadhan, and AFOSR award FA9550-25-1-0112.
Salil Vadhan: Supported by a Simons Investigators Award.

1 Introduction

1.1 Background

Suppose that one wishes to run an algorithm on an Erdős–Rényi random graph $\mathcal{G}(n,p)$. There are exponentially many graphs in the support of $\mathcal{G}(n,p)$, so even storing a graph requires resources polynomial in n. A common approach for derandomizing such an algorithm is to replace $\mathcal{G}(n,p)$ with some random looking distribution $\mathcal{G}_n$ whose support is much smaller (for example, $\text{poly}(n)$), so that all the properties of random graphs on which the algorithm relies are still preserved. This line of work was initiated by Goldreich, Goldwasser, and Nussboim [16], who studied pseudorandom graphs that are indistinguishable from $\mathcal{G}(n,p)$ by any oracle machine that makes $\text{poly}(\log n)$ adjacency queries to a graph. Goldreich et al. considered several properties of random graphs, such as connectedness and Hamiltonicity, and presented pseudorandom graph constructions that preserve these properties while having support polynomial in n.

Alon and Nussboim [4] studied the specific strategy of using k-wise independence for sampling a pseudorandom graph. These can be sampled with support size $n^{\Theta(k)}$ and stored and accessed with $\widetilde{O}(k \log n)$ space and time. Similarly to [16], [4] show that many typical properties and thresholds that are enjoyed by truly random graphs *still* hold true with k-wise independent sampling for $k = O(\log n)$: for instance, edge connectivity, vertex connectivity, jumbledness, and the existence of matchings all mimic the behavior of truly independent sampling.

We continue this line of study by focusing on random subsampling of edges in graphs: instead of looking at $\mathcal{G}(n,p)$, we start with a given graph and randomly take each of its edges in our sample with some fixed probability, getting a sparser subgraph. This is a fundamental technique which underlies a huge portion of modern graph algorithms, including sparsification and its many applications [8, 28], coloring [5], vast arrays of sublinear algorithms [25], and many more. However, for these graph algorithms, the gold standard is often the design of *deterministic* algorithms, thereby bypassing any uncertainty in the algorithm's output. Motivated by this, the focus of one line of study, including this work, is on the ability to *derandomize* graph sampling.

Subsequent works have studied the feasibility of k-wise independent subsampling in arbitrary graphs: as an example, the work of Doron, Murtagh, Vadhan, and Zuckerman [11] studied the ability of k-wise independent sampling to produce *spectral sparsifiers* of graphs, again showing that mild independence (in fact, $k = O(\log(n))$) suffices for preserving the graph's spectrum.

However, these aforementioned results rely on a shared insight: namely that bounded independence sampling suffices in good spectral expanders for concentration of the *eigenvalues* of a graph's Laplacian. Alon and Nussboim [4] leverage the fact that a complete graph is a good expander, and therefore has eigenvalues well-separated from 0, and Doron et al. [11] assign different sampling probabilities to each edge proportional to their effective resistances, which was a procedure known to preserve graph's eigenvalues with fully independent sampling [27].

Thus, in our work we seek to understand:

> *Does k-wise independent subsampling for small k also preserve* combinatorial *properties of graphs that are not good expanders?*

By "combinatorial" properties, we mean those where we do not know how to use the spectrum directly to analyze fully independent subsampling.

As we shall see, we show that for certain properties the answer is strongly affirmative. In the following section, we discuss these results in more detail and subsequently explain how they lend themselves to new explicit derandomized graph algorithms.

1.2 Our Graph Sampling Results

As an illustration of the challenges we face, consider a graph $G = (V, E)$ on n vertices with minimum cut $\lambda \approx 100 \log(n)$. Karger's [18] cut-counting bound establishes that in such a graph, the number of cuts of size $\leq \alpha\lambda$ is at most $n^{2\alpha}$. An immediate corollary of this is that in such a graph G, if one samples each edge independently with probability $1/2$, then the resulting graph is connected with probability at least $1 - 1/\text{poly}(n)$. To see why, we bin our cuts to those of size in the interval $[\lambda, 2\lambda)$, $[2\lambda, 4\lambda)$, etc. Each cut in the bin $[\alpha\lambda, 2\alpha\lambda)$ has at least one edge in the surviving sample of edges with probability $1 - 1/2^{\alpha\lambda} = 1 - 1/n^{100\alpha}$. We can then take a union bound over all $\leq n^{4\alpha}$ cuts in this bin and then a further union bound over all $\leq n$ bins to conclude that *every* cut has at least one surviving edge in the sample with high probability, and thus the graph is connected.

As mentioned above, a natural approach to derandomizing the above property is to use k-wise independent sampling. Unfortunately, if we revisit the argument above, we can see that an essential step in arguing that the graph is connected under random subsampling is to take a union bound over an *exponential* number of cuts in the graph. This relies on having "success" probabilities for individual cuts that are as large as $1 - 2^{-(n-1)}$, since there are 2^{n-1} cuts in the graph. In general, no pseudorandom algorithm with a sample space smaller than 2^{n-1} can guarantee such high success probabilities, since for such an algorithm, the individual probabilities are multiples of $2^{-(n-1)}$.

Furthermore, a graph having a minimum cut of size $100 \log(n)$ provides *no meaningful bound* on its expansion or the effective resistances of its underlying edges, meaning that the techniques of [4] and [11] do not directly lend themselves to this setting. Thus, it may seem that we are unable to conclude this relatively benign connectivity property under bounded-independence sampling.

Despite the appearance of the above barriers, we show that for $k = O(\log(n))$, k-wise independent sampling still yields connected graphs with high probability.

▶ **Theorem 1.1.** *Let $G = (V, E)$ be a graph with m edges and minimum cut $\lambda \geq \kappa \log(m)$ for some absolute constant κ. Let $\widetilde{G}$ denote the result of subsampling edges of G in a $2\kappa \log(m)$-wise independent manner with marginals $1/2$. Then, $\widetilde{G}$ is connected with probability $\geq 1 - 1/\text{poly}(m)$.*

This result serves as a strong generalization of some of the results of [4]. Indeed, [4] studies which properties of *the complete graph* are preserved under k-wise independent subsampling. Naturally, working with such a structured graph enables simpler analysis. Nevertheless, in our much less structured regime where graphs only have a lower bound on their minimum cut, we are still able to show that $O(\log(m))$-wise independent sampling suffices for preserving connectivity.

Sampling from a $O(\log m)$-wise independent distribution requires $\Theta(\log^2(m))$ random bits; indeed the support must be of size $2^{\Omega(\log^2(m))}$ [3]. Thus, as an alternative, we construct another sampling procedure that maintains connectivity while having almost-polynomial support size:

▶ **Theorem 1.2.** *There exists an explicit distribution Y on $\{0,1\}^m$ that can be sampled using $O(\log m \cdot \log\log m)$ random bits and has marginals $\leq 1/2$ such that the following holds. For every graph $G = (V, E)$ with m edges and minimum cut $\geq \kappa \log(m)$ for some absolute constant κ, subsampling the edges of G using Y yields a connected subgraph with probability $1 - 1/\text{poly}(m)$.*

In a similar manner, we also study *cycle-free-ness* when randomly sampling edges with large girth. In more detail, consider a graph $G = (V, E)$ whose shortest cycle is of length $\geq 100 \log(n) = \lambda$. A bound due to Subramaniam [29] (and rediscovered in several other works, including [13]) states that in such a graph, the number of cycles of length $\leq \alpha \cdot \lambda$, is bounded by $n^{2\alpha}$. Thus, if one samples the edges in this graph uniformly at rate $1/2$, the exact same argument as in the cut setting above will show that the resulting sample of edges is *cycle free* with high probability. However, this argument again relies on exponential concentration in the tail, and thus does not immediately translate into a proof of the same property holding under bounded independence. Despite this, we are able to show that bounded independence suffices:

▶ **Theorem 1.3.** *Let $G = (V, E)$ be a graph with m edges and minimum cycle length $\lambda \geq \kappa \log(m)$ for some absolute constant κ. Let $\widetilde{G}$ denote the result of subsampling edges of G in a $(1/m^{200})$-almost $2\kappa \log(m)$-wise independent manner with marginals $1/2$. Then, $\widetilde{G}$ is cycle free with probability $\geq 1 - 1/\text{poly}(m)$.*

Unlike Theorem 1.1, this theorem allows for the use of an *almost* $O(\log m)$-wise independent distribution, which can be sampled using $O(\log m)$ random bits.

Together, these results show that bounded-independence sampling *does suffice* for preserving basic combinatorial properties in weakly-structured graphs (i.e., non-expanders). As we elaborate upon more in the Technical Overview (Section 1.4), our results rely on very careful analyses of graph structure. Furthermore, this improved understanding of graph sampling under bounded-independence immediately lends itself to better deterministic parallel algorithms for finding matroid bases.

1.3 Applications to Parallel Matroid Algorithms

1.3.1 Matroid Background

Matroids are fundamental and ubiquitous objects in combinatorial optimization, generalizing a number of basic structures like spanning forests in graphs, and bases of vector spaces. More formally, a matroid is modelled as a set system $\mathcal{M} = (E, \mathcal{I})$, where E is the *ground set*, consisting of m elements, and $\mathcal{I} \subseteq 2^E$ is the *set of independent sets*, satisfying:

1. $\emptyset \in \mathcal{I}$ (non-empty property).
2. If $S \in \mathcal{I}$ and $S' \subseteq S$, then $S \in \mathcal{I}$ (downward closure property).
3. If $S, T \in \mathcal{I}$ and $|S| > |T|$, then $\exists e \in S \setminus T : T \cup \{e\} \in \mathcal{I}$ (exchange property).

Within the study of matroids, a critical notion is that of a *basis*, or namely a set $S \in \mathcal{I}$ that is *maximal* under inclusion. Note that because of the exchange property above, we know that for a given matroid $\mathcal{M} = (E, \mathcal{I})$, all bases of $\mathcal{M}$ are of the same size.

For instance, one can take the ground set E to be the set of edges in a graph $G = (V, E)$, and let a set of edges $S \subseteq E$ be independent if and only if S is *cycle-free* in the graph G. This yields the family of so-called *graphic matroids*, with a maximal independent set (i.e., basis) being a spanning forest of G.

1.3.2 Parallel Basis Finding

Despite their prevalence in combinatorial optimization (see e.g., [12, 17, 15, 24, 6, 9]), our understanding of matroids is incomplete, even in basic directions. One such direction is in the study of *parallel algorithms* for finding matroid bases. In this model, first introduced by Karp, Upfal, and Wigderson [19, 20], one wishes to find a basis of some matroid $\mathcal{M} = (E, \mathcal{I})$.

However, because the number of matroids on n elements grows as 2^{2^n} [7], it is too expensive to store the entire description of the matroid. Instead, the algorithm is given access to the matroid only in the form of an *independence oracle*; i.e., a function $\mathrm{Ind} : 2^E \to \{0,1\}$, such that $\mathrm{Ind}(S) = \mathbf{1}[S \in \mathcal{I}]$. Under this form of access, the works of Karp, Upfal, and Wigderson [19, 20] asked:

> *How many adaptive rounds of independence queries are required to find a basis of an arbitrary matroid, when each round is allowed only* $\mathrm{poly}(n)$ *many queries to the independence oracle?*

In the seminal work [19], it was proven that in the setting of arbitrary matroids, there is a *deterministic* algorithm which finds a basis of any matroid in just $O(\sqrt{n})$ rounds, and that there is a lower bound of $\Omega(n^{1/3})$ many rounds for this task. Despite providing a narrow gap for the parallel complexity of this basic problem, the ensuing four decades saw no improvements to these bounds. Only recently, in the work of Khanna, Putterman, and Song [21, 23], were these bounds improved, with the establishment of an $\widetilde{O}(n^{3/7})$ round *randomized* algorithm for finding bases in arbitrary matroids. Nevertheless, the true complexity of this problem is yet to be established.

Due to the complex structure of arbitrary matroids, the work of [19] proposed studying the parallel complexity of *concrete classes* of matroids. Among these, the most notable class studied by [19] is the class of *graphic matroids*:

▶ **Definition 1.4.** *Given a graph $G = (V, E)$, the* graphic matroid *induced by G is the matroid $\mathcal{M} = (E, \mathcal{I})$ such that for a set $S \subseteq E$, $S \in \mathcal{I}$ if and only if S is cycle-free.*

For this class of matroids, [19] provided deterministic algorithms that, on graphs with m edges, achieve either (a) for any constant $d \in \mathbb{N}$, $m^{1/d}$ rounds and $m^{O(d)}$ queries per round, or (b) $O(\log(m))$ rounds with $m^{O(\log(m))}$ queries per round. This was later improved in the work of Khanna, Putterman, and Song [22], which provided an algorithm that finds bases of graphic matroids with only $\mathrm{poly}(m)$ queries per round and $O(\log(m))$ many rounds.

Note that, as mentioned in their work, the algorithms of [22] can be derandomized *non-explicitly*, in an argument akin to Adleman's proof of $\mathbf{BPP} \subseteq \mathbf{P/poly}$ [1]. This of course answers the natural question of the deterministic "query complexity" of finding a matroid basis, but does not yield an efficiently implementable uniform algorithm.

1.3.3 Our Results

Our first result uses Theorem 1.3 to deterministically find bases in *graphic matroids*. In this regime, we show the following:

▶ **Theorem 1.5.** *There is a uniform, deterministic algorithm that, for any graphic matroid on m elements, finds a basis in $O(\log m \cdot \log\log m)$ many rounds of $\mathrm{poly}(m)$ queries.*

This theorem shows that the result of [22] *can* be explicitly derandomized with only a minor increase in the number of rounds (from $\log(m)$ to $\log(m)\log\log(m)$), while still retaining a polynomial number of queries per round. The proof of this result relies intimately on Theorem 1.3 (in fact, even on a strengthening of Theorem 1.3).

Next, as an application of Theorem 1.2, we show that basis finding in *cographic* matroids can also be derandomized. A set of edges $S \subseteq E$ in a cographic matroid for a graph G is defined to be independent if and only if S does not completely contain any non-empty cut C of G, and so a basis of a cographic matroid is a set of edges that is the *complement* of a spanning forest.

▶ **Theorem 1.6.** *There is a uniform, deterministic algorithm that, for any cographic matroid on m elements, finds a basis in only $O(\log m \cdot \log\log m)$ many rounds of $m^{O(\log\log(m))}$ queries.*

Note that, prior to this work, there was no non-trivial uniform, deterministic algorithm for basis finding in cographic matroids. The best algorithm was only known to use $O(\sqrt{m})$ rounds and poly(m) queries per round - due to [19] for *arbitrary matroids* (thus including cographic ones as a special case). We view Theorem 1.6 as further evidence that one can likely obtain polylog(m) rounds and poly(m) queries, though we leave this as an open question.

As we elaborate upon in the next section, both Theorem 1.5 and Theorem 1.6 rely on a new algorithmic framework for finding bases of matroids which, unlike the algorithms of [22], uses both deletion and contraction of elements. However, to remove the randomness in this algorithmic framework, our results rely on the concrete derandomization results for basic graph sampling tasks that we discussed above.

1.4 Technical Overview

In this overview, we discuss some of the intuition that underlies both our derandomized graph sampling results as well as our improved explicit deterministic matroid algorithms.

1.4.1 Derandomizing Graph Sampling

We start by providing intuition for Theorem 1.1. Before doing so, we briefly recap the discussion in Section 1.2. In this setting, we are given a graph $G = (V, E)$ with minimum cut λ of size $\geq \kappa \log(m)$ for some constant κ. Our goal is to subsample the edges of this graph at rate $1/2$ using a *bounded-independence* distribution while still ensuring that the resulting sampled graph is connected with high probability. As mentioned in Section 1.2 however, a simple union bound over all of the cuts in the graph *does not* suffice for arguing connectivity when using bounded-independence distributions. Indeed, there are 2^{n-1} cuts in a graph on n vertices, and thus any union bound requires that most such cuts have at least one surviving edge with probability $1 - \Omega\left(2^{-n}\right)$; a probability which is too close to 1 to be possible under k-wise independent sampling with $k = o(n)$.

For comparison, [4] and [11] have studied bounded-independence sampling for small k (generally, $k = O(\log(n))$) and showed that in these settings, many spectral properties of graphs which are good expanders can be preserved. Here, spectral properties of a graph $G = (V, E)$ refer to properties of the graph's Laplacian $L_G = \sum_{e=(u,v)\in E} w_e \cdot \chi_e \chi_e^T$, where $\chi_e \in \mathbb{R}^n$ is the vector such that $\chi_e = \mathbf{1}_u - \mathbf{1}_v$ when $e = (u, v)$. In this direction, [11] provides a more general result than [4]: indeed, it follows from [11] that if the *leverage scores* in a graph G are all smaller than $\approx \frac{\varepsilon^2}{2\log(n)}$, then $O(\log(n))$-wise independent sampling with marginals $1/2$ yields a *spectral sparsifier* of the graph G. In this context, the leverage score of an edge $e = (u, v)$ is defined as $\ell(u, v) = w_e \cdot \chi_e^T L_G^\dagger \chi_e$ and provides a measure of how important the edge (u, v) is for preserving connectivity between u and v. A spectral sparsifier of a graph G is simply a subgraph $\widetilde{G}$ such that $(1-\varepsilon)L_G \preceq L_{\widetilde{G}} \preceq (1+\varepsilon)L_G$, where $\preceq$ is the Loewner order on PSD matrices ($A \preceq B$ if $x^T A x \leq x^T B x$ for all x). If $\widetilde{G}$ is a spectral sparsifier of G, then this implies that for every $x \in \{0, 1\}^n$, $x^T L_{\widetilde{G}} x \in (1 \pm \varepsilon) x^T L_G x$, which means that $x^T L_{\widetilde{G}} x = 0$ if and only if $x^T L_G x = 0$. This means that $\widetilde{G}$ and G will have the same number of connected components (since these equal the multiplicity of 0 as an eigenvalue of the Laplacian), and in fact the connected components will be identical.

This provides a natural first attempt towards showing that $O(\log(m))$-wise independent sampling in our graph G preserves connectivity. I.e., can we try to argue the *stronger* property that the resulting sampled edges constitute a *spectral sparsifier* of G? Unfortunately,

this turns out not to be the case. This is because a graph having a large minimum cut is very strongly *not* sufficient for the graph to have small leverage scores (see [14] for more discussion). Indeed, even if we tried sampling the edges of G *uniformly* at random, the resulting graph would not be guaranteed to be a spectral sparsifier with high probability.

To overcome this, we rely on a key observation: the graph G that we are working with so far is *unweighted*. But, edge weights *can* alter the behavior of the graph's Laplacian, and in turn, the leverage scores of the edges. Thus, one may wonder: is it possible to re-weight the edges of the graph such that all the leverage scores become smaller? Indeed, one of our key contributions is to show that this is true, and in doing so, we provide a new connection between leverage scores and a graph's minimum cut:

▶ **Theorem 1.7.** *Given an unweighted graph $G = (V, E)$ with minimum cut c, there exists a weighting $w : E \to \mathbb{R}_{\geq 0}$ such that for the weighted graph $G' = (V, E, w)$ and for each $e \in E$, the leverage score of e in G' is $\ell(e) \leq O(1/c)$.*

Moreover, the converse is also true: if there exists a weighting such that all leverage scores are at most $1/c$, the minimal cut of the original graph is $\geq c$.

With this theorem in hand, we are then immediately able to show that connectivity is preserved whenever our graph G has minimum cut $\geq \kappa \log(m)$: indeed, given the graph $G = (V, E)$, we implicitly analyze the *re-weighted* graph $\hat{G}$. On $\hat{G}$, we sample each edge using our $O(\log(m))$-wise independent distribution, and are guaranteed via [11] that the resulting sampled edges are a spectral sparsifier of $\hat{G}$, which in particular means that the sampled edges form a connected graph. Then, one must only observe that the sampling procedure on G and $\hat{G}$ is identical, as both have the same set of edges.

Proving Theorem 1.7 is more subtle; our proof relies on a "bounded effective resistance diameter" decomposition theorem of [2], along with a careful, recursive weight assignment scheme. We omit the details from the technical overview for brevity, and direct the reader to Section 3.1 for more details.

In a separate direction, our proof of Theorem 1.3 foregoes a spectral argument entirely. Indeed, in this setting where we must argue *cycle-freeness*, we instead show that the existence of *any cycles* in the graph can instead be modeled via the existence of *short paths* in the graph. This allows us to transition from an event space of nearly-exponential size (i.e., the set of all cycles) to instead work with an event space of only polynomial size (pairs of vertices with short paths between them). We present this argument in Section 4.

The proof Theorem 1.2 relies on different techniques. We construct our distribution Y on $\{0,1\}^m$ by combining $O(\log m)$ independent copies of any distribution Y_0 on $\{0,1\}^{m \times \log\log m}$ that fools read-once DNF formulas on $\widetilde{O}(\log m)$ (out of the m possible) variables. By the results of De, Etesami, Trevisan, and Tulsiani [10], such a distribution Y_0 can be sampled using $O(\log\log m)$ random bits, meaning that we can generate Y using $O(\log m \cdot \log\log m)$ random bits. Interestingly, the way we combine the copies of Y_0 to set Y is also using a read-once DNF formula, namely the Tribes function.

Together, these results provide a robust toolkit for reasoning about graph properties under bounded-independence sampling when the underlying graph is not a good expander. As we shall see below, this has direct applications to derandomized matroid basis computation.

1.4.2 Derandomized Matroid Basis Finding

Once we have our improved derandomized graph sampling results, there are still barriers towards implementing better explicit parallel basis finding algorithms. To illustrate these barriers, we first revisit the algorithm of [22].

Indeed, let us consider the setting of graphic matroids; recall that here there is an underlying graph $G = (V, E)$, and the goal of the algorithm is to find a *spanning forest* of G. The algorithms in this setting only have access to an *independence oracle*, meaning that the algorithm can query a set $S \subseteq E$ of edges, and the oracle reports whether the set S has any cycles. The intuition for the algorithm of [22] is that each round of the algorithm *deletes* more edges from the graph without altering the connected components of the graph. Then, after $O(\log(n))$ many rounds of computation, [22] shows that the graph has no cycles remaining, and thus the leftover edges must be a spanning forest of the original graph G. To implement this argument more carefully, [22] relies on two key invariants:

1. After i rounds of the algorithm, the remaining edges $E^{(i)} \subseteq E$ induce the same connected components as E.
2. After i rounds of the algorithm, there are no cycles in $E^{(i)}$ of length $\leq 1.01^i$.

Clearly then, one can see that after $O(\log(n))$ rounds, the remaining edges constitute a spanning forest.

To actually ensure these invariants hold, [22] relies on random sampling of edges. Indeed, in an iteration i where the minimum cycle length in $E^{(i)}$ is $\lambda = 1.01^i$, [22] shows that *uniformly random sampling* of edges can be used to isolate any near-minimum length cycle. Formally, [22] shows:

▶ **Lemma 1.8** ([22]). *Let $G = (V, E)$ be a graph with m edges and minimum cycle length λ. Let $C \subseteq E$ be a cycle in G of length $\leq 1.01\lambda$. Now, let $\widetilde{G}$ be the result of uniformly randomly sampling the edges of G at rate $p = 1/m^{100/\lambda}$. Then, $C \subseteq E$ is the* unique *surviving cycle in $\widetilde{G}$ with probability at least $1/\mathrm{poly}(m)$.*

With such a lemma in hand, [22] performs random sampling a large polynomial number times, ensuring that every single near-minimum length cycle is the unique surviving cycle in some sampled graph. This then enables [22] to recover the identities of *all* near-minimum length cycles (and in so doing, also enables their removal without altering connectivity).

Our goal is to adopt this algorithmic framework, but to use bounded-independence sampling instead of truly random sampling. Ultimately, this then allows us to *enumerate* all possible samples of edges, thereby obtaining explicit, deterministic results.

Unfortunately, directly using our derandomized graph sampling results *does not* work inside this algorithmic framework. There are two primary reasons why:

1.4.2.1 Unique Cycle Survival

First, as currently stated, Theorem 1.3 only governs the probability that a sampled graph is *cycle-free*, *not* the probability that there is a single, unique cycle which survives the bounded-independence sampling. Fortunately, Theorem 1.3 can be modified to ensure unique cycle survival in the regime where the minimum cycle length is $\lambda = \Theta(\log(m))$. To see why, fix a single cycle C of length $\leq 1.01\lambda$. For this cycle, if one uses (almost) k-wise independent sampling with $k \geq 10\lambda$, then C's survival probability under bounded-independence sampling is in fact *the same* as under a uniformly random distribution. Now, conditioned on C's survival, we can re-invoke Theorem 1.3 on the remaining graph to ensure that *no other cycles* survive sampling. This argument requires care, and appears in Section 4.2.

1.4.2.2 Large Cycle Lengths

The second barrier is that, as stated, Theorem 1.3 requires a minimum cycle length of at least $\kappa \log(m)$, along with sampling at rate $1/2$. Not only this, but in the previous barrier, the argument we just discussed relies on using k-wise (almost) independent sampling for $k \geq 10\lambda$, where λ is the current minimum cycle length.

Unfortunately, the algorithmic framework of [22] requires a spectrum of different cycle lengths and sampling rates, starting with a minimum cycle length of 1, and slowly increasing as edges are deleted between rounds. Thus, there are even rounds where the minimum cycle length $\lambda = \Omega(m)$, which in our above argument requires $\Omega(m)$-wise independent sampling, which is far too costly.

To overcome this barrier, we employ a much more careful analysis than [22]: rather than continuing to remove short cycles until the minimum cycle length is $> m$, we instead invest $O(\log\log(m))$ rounds until the minimum cycle length becomes $\approx \kappa \log(m)$. Note that in these rounds where the minimum cycle length is $o(\log(m))$, we even require a modification to Theorem 1.3 which shows that one can sample *at smaller marginal rates* (instead of $1/2$) while still ensuring unique cycle survival. We then progressively alter these sampling rates until the minimum cycle length becomes $\approx \kappa \log(m)$.

Once the minimum cycle length becomes $\approx \kappa \log(m)$, instead of trying to enumerate short cycles in the graph, we instead *directly invoke* Theorem 1.3 to find *large sets of cycle-free edges* (i.e., $\Omega(m)$ many edges without any cycles). Once we have such a set of edges, we then *commit* to including these edges in our final spanning forest. Thus, from a graph on n vertices, we now have collected $\Omega(m)$ of the edges we need for our ultimate spanning forest.

The key invariant now is that every $O(\log\log(m))$ rounds, we recover an $\Omega(1)$ fraction of the remaining edges we need for our spanning forest. Thus, after only $O(\log(m)\log\log(m))$ rounds, we recover a spanning forest of our starting graph.

We use the same strategy in the cographic setting. Namely, within each recursive iteration we invest $O(\log\log(m))$ rounds to eliminate small cuts, so that the minimum cut becomes $\approx \kappa \log(m)$ (to achieve this, we rely on a modification of Theorem 1.2 that allows sampling at smaller marginal rates). We then directly invoke Theorem 1.2 to find a large independent set and commit to including it in the basis, moving to the next iteration.

1.5 Organization

We start with preliminaries in Section 2. In Section 3 and Section 4, we provide tight guarantees on how graph structure behaves under bounded independence sampling: Section 3 shows that graphs with logarithmic min-cut remain connected under bounded-independence sampling, while Section 4 shows that graphs with large minimum cycle length become cycle-free.

In the full version, we present a general framework that underlies our basis finding algorithms for both graphic and cographic matroids. We then use the cycle-freeness characterization to design near-optimal explicit, deterministic parallel algorithms for finding bases of graphic matroids, and use the aforementioned connectedness guarantee to provide explicit, deterministic parallel algorithms for finding bases of cographic matroids.

2 Preliminaries

We begin with providing tools and definitions that are necessary for the remainder of this paper.

An *unweighted undirected graph* is defined as $G = (V, E)$, where V is the set of vertices and E is the set of edges. We always use n, m to be the number of vertices and edges of a graph, respectively: $n = |V|, m = |E|$. A *weighted* graph $G = (V, E, w)$ also has a function $w : E \to \mathbb{R}^+$, weighting the edges. For any subset of edges $A \subseteq E$, $w(A)$ denotes the sum of the weights of all edges in A. For any $v \in V$, $\delta(v) \subseteq V$ denotes the set of *neighbors* of v in G. For any subset of vertices $V' \subseteq V$, $G[V'] = (V', E', w')$ denotes the *subgraph of G induced*

by V', while $E(V') \subseteq E$ denotes the set of edges with both endpoints in V'. We say that a set of edges $C \subseteq E$ is a *cut* in G if and only if there is a set $S \subseteq V$ such that C is the set of edges between S and $\bar{S}$.

For any subset of vertices $S \subseteq V$, G/S denotes the result of *contracting* G on a set S, i.e. identifying all the vertices of S. Specifically, $G/S = ((V \setminus S) \cup \{s^*\}, E')$, where $E' = \{(u,v) : (u,v) \in E \text{ and } u,v \notin S\} \cup \{(u,s^*) : (u,s) \in E \text{ for some } s \in S\}$. Note that G/S is a multigraph: if u is connected to multiple edges in S, we keep all these edges as parallel edges in G/S.

Importantly, contracting a graph on a subset of vertices $S \subseteq V$ does not decrease its minimum cut, since it only limits the set of all cuts to those that do not cut through S.

We say that $f : \mathbb{N} \to \mathbb{N}$ is $\text{poly}(n)$ if there exist $n_0, c \in \mathbb{N}$ such that $\forall n \geq n_0$, $f(n) \leq n^c$.

Finally, we use $\tilde{O}$ notation to omit polylogarithmic factors: $\tilde{O}(f(n)) = O(f(n) \cdot \text{poly}(\log n))$.

2.1 Bounded-Independence Sampling

We start by recalling the notion of a δ-almost k-wise independent distribution:

▶ **Definition 2.1.** *We say that a distribution* $X = (x_1, \dots x_n)$ *is* δ-almost k-wise independent *with marginals* p *if for all subsets* $S \subseteq [n], |S| \leq k$, *we have*

$$d_{\text{TV}}(U(S), X(S)) \leq \delta,$$

where d_{TV} *is the* total variation distance *and* $U(S)$ *denotes the uniform distribution (with marginals* p*) over the set* S. *When* $\delta = 0$, *i.e.* $X(S) = U(S)\ \forall S : |S| \leq k$, *we say that* X *is* k*-wise independent.*

The work of Naor and Naor [26] explicitly constructed such δ-almost k-wise independent random variables of small size:

▶ **Theorem 2.2** ([26])**.** *There is an explicit construction of a* δ*-almost* k*-wise independent distribution* $X = (x_1, \dots x_n)$ *with marginals* $1/2$ *and seed length* $O(k + \log\log(n) + \log(1/\delta))$.

By "explicit" construction of a distribution X on $\{0,1\}^n$, we mean that there is a polynomial time algorithm $G : \{0,1\}^r \to \{0,1\}^n$ (where r is the *seed length* or the *number of random bits* used to generate X), such that $X = G(U_r)$.

When $\delta = 0$ (exact k-wise independence) we have the following space bounds:

▶ **Theorem 2.3** ([30])**.** *There is an explicit construction of a* k*-wise independent distribution* $X = (x_1, \dots x_n)$ *with marginals* $1/2$ *and seed length* $O(k \log(n))$.

The following reduction allows us to construct bounded-independence distributions with arbitrarily small marginal probabilities:

Algorithm 2.1 SampleLowerMarginals.

input : $n, k \in \mathbb{N}, \delta \geq 0, p$ s.t. $\log(1/p) \in \mathbb{Z}$;
$(y_1, \dots y_{n\log(1/p)})$ sampled from a δ-almost $k\log(1/p)$-wise independent distribution Y with marginals $1/2$.

output : $X = (x_1, ..., x_n)$, where $x_i = \prod_{j=(i-1)\log(1/p)+1}^{i\log(1/p)} y_j$ for every $i \in [n]$.

▶ **Lemma 2.4.** *For any $n, k \in \mathbb{N}$, $\delta \geq 0$, p such that $\log(1/p)$ is an integer, and δ-almost $k\log(1/p)$-wise independent distribution $Y = (y_1, \ldots y_{n\log(1/p)})$ with marginals $1/2$, the distribution $X = (x_1, \ldots x_n)$ constructed using Algorithm 2.1 is δ-almost k-wise independent with marginals p.*

The proof of this lemma is deferred to the full version of the paper.

The support size of the distribution X obtained from Algorithm 2.1 is the same as the support size of the original distribution Y with marginals $1/2$. Hence, for $\delta > 0$, using Theorem 2.2 with Algorithm 2.1 we get a δ-almost k-wise independent distribution with marginals p and seed length $O(k\log(1/p) + \log\log(n\log(1/p)) + \log(1/\delta))$. Similarly, for $\delta = 0$, using Theorem 2.3 with Algorithm 2.1 we get a k-wise independent distribution with marginals p and seed length $O(k\log(1/p)\log(n\log(1/p)))$.

2.2 Electric Flow and Effective Resistance

▶ **Definition 2.5** (Laplacian Matrix). *Given an undirected weighted graph $G = (V, E, w)$ on n vertices, the* Laplacian matrix $\mathbf{L}_G \in \mathbb{R}^{n\times n}$ *of G is defined as:*

$$\mathbf{L}_G = \mathbf{D}_G - \mathbf{A}_G,$$

where $\mathbf{D}_G \in \mathbb{R}^{n\times n}$ is the diagonal matrix of weighted degrees of G, and $\mathbf{A}_G$ is the weighted adjacency matrix of G: $(\mathbf{A}_G)_{ij} = (\mathbf{A}_G)_{ji} = w_{ij}$.

It is easy to verify that for any graph G, $\mathbf{L}_G$ is positive semidefinite. Indeed, for any $\mathbf{x} \in \mathbb{R}^n$, the Laplacian quadratic form is: $\mathbf{x}^T\mathbf{L}_G\mathbf{x} = \sum_{(u,v)\in E} w(u,v)\cdot(\mathbf{x}(u) - \mathbf{x}(v))^2 \geq 0$ since the weights are non-negative.

Since $\mathbf{L}_G$ is symmetric, it has a basis of eigenvectors. Hence, we use $\lambda_1, \lambda_2, ..., \lambda_n$ to denote the eigenvalues of $\mathbf{L}_G$ in increasing order ($\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$). Since $\mathbf{L}_G\mathbf{1}^n = 0$, $\lambda_1 = 0$ for any graph G. It can further be verified that $\lambda_2 = 0$ iff G is not connected.

▶ **Definition 2.6** (Spectral Approximation). *For $\varepsilon \geq 0$, a graph H is an* ε-spectral approximation *of a graph G if for all $\mathbf{x} \in \mathbb{R}^n$ we have that $\mathbf{x}^T\mathbf{L}_H\mathbf{x} \in (1 \pm \varepsilon)\cdot\mathbf{x}^T\mathbf{L}_G\mathbf{x}$. In this case we write $\mathbf{L}_H \cong_\varepsilon \mathbf{L}_G$.*

▶ **Definition 2.7** (Network Flow). *Given a graph $G = (V, E, w)$, some external flows $\mathbf{i}_{ext} \in \mathbb{R}^n$, and an arbitrary orientation of edges $E^\pm$, a* flow *is a function $\mathbf{f} \in \mathbb{R}^{E^\pm}$ such that $\mathbf{f}(u,v) = -\mathbf{f}(v,u)\ \forall (u,v) \in E^\pm$ and for any $v \in V$:*

$$(\mathbf{i}_{ext})_v + \sum_{u\in\delta(v)} \mathbf{f}(u,v) = 0 \qquad \text{(flow conservation)}$$

The flow conservation law simply implies that no flow disappears or is created at any vertex. In contrast to arbitrary network flows, an electric flow also has to obey Ohm's law:

▶ **Definition 2.8** (Electric Flow). *Given a graph $G = (V, E, w)$, external flows $\mathbf{i}_{ext} \in \mathbb{R}^V$, and an arbitrary orientation of edges $E^\pm$, a network flow $\mathbf{f} \in \mathbb{R}^{E^\pm}$ (as in Definition 2.7) is an* electric flow *if there exists a potential function $\mathbf{p} \in \mathbb{R}^V$ such that $\forall (u,v) \in E$:*

$$\mathbf{f}(u,v) = w(u,v)(\mathbf{p}(u) - \mathbf{p}(v)) \qquad \text{(Ohm's law)}$$

An important property that distinguishes an electric flow from just an arbitrary network flow is that it minimizes the electrical energy:

▶ **Definition 2.9** (Electrical Energy). *Given a graph $G = (V, E, w)$ and some network flow $\mathbf{f} \in \mathbb{R}^{E^{\pm}}$ (as in Definition 2.7) in this graph, the* electrical energy *of $\mathbf{f}$ is defined as:*

$$\mathcal{E}(\mathbf{f}) = \sum_{e \in E^{\pm}} \frac{\mathbf{f}(e)^2}{w(e)}$$

▶ **Fact 2.10.** *Given a graph $G = (V, E, w)$ and external currents $\mathbf{i}_{ext} \in \mathbb{R}^V$, an electric flow $\mathbf{f} \in \mathbb{R}^{E^{\pm}}$ is the unique network flow (with the given external currents) minimizing $\mathcal{E}(\mathbf{f})$.*

▶ **Definition 2.11** (Effective Resistance). *Given a weighted undirected graph $G = (V, E, w)$, the* effective resistance *between $u, v \in V$ is the potential difference between u and v induced by the unit u-v electric flow (i.e. the unique electric flow induced by setting $\mathbf{i}_{ext}(u) = 1$, $\mathbf{i}_{ext}(v) = -1$, and $\mathbf{i}_{ext}(s) = 0$ for $s \notin \{u, v\}$):*

$$\mathrm{Reff}_G(u, v) = \mathbf{p}(u) - \mathbf{p}(v)$$

▶ **Fact 2.12** (Thomson's Principle). *Given a graph $G = (V, E, w)$ and a unit u-v electric flow $\mathbf{f}_{uv} \in \mathbb{R}^{E^{\pm}}$, $\mathrm{Reff}_G(u, v) = \mathcal{E}(\mathbf{f}_{uv})$.*

▶ **Definition 2.13** (Effective Resistance Diameter). Effective resistance diameter *of a graph $G = (V, E, w)$ equals the maximum effective resistance among all pairs of vertices:*

$$\mathrm{R}_{\mathrm{diam}}(G) = \max_{u,v \in V} \mathrm{Reff}_G(u, v)$$

▶ **Definition 2.14** (Leverage Score). *Given a graph $G = (V, E, w)$, the* leverage score *of an edge $(u, v) \in E$, denoted as $\ell(u, v)$, is the ratio between the effective resistance between its endpoints $\mathrm{Reff}_G(u, v)$ and the actual resistance of an edge $1/w(u, v)$:*

$$\ell(u, v) = w(u, v) \cdot \mathrm{Reff}_G(u, v)$$

▶ **Fact 2.15.** *Consider a connected graph $G = (V, E, w)$. If we sample a spanning tree of G so that the probability of sampling a given tree is proportional to the product of the weights of edges in it, then the probability that a particular edge $(u, v) \in E$ is included in the sampled tree equals $\ell(u, v)$.*

2.3 Graph Clustering using Effective Resistance

Alev et al. [2] show that any graph can be partitioned into components with small effective resistance diameter:

▶ **Theorem 2.16** ([2]). *Given a weighted graph $G = (V, E, w)$, and a large enough parameter $\delta > 1$, there is an algorithm with time complexity $\tilde{O}\left(m \cdot n \cdot \log\left(\frac{w(E)}{\min_e w(e)}\right)\right)$ that finds a partition $V = \bigcup_{i=1}^{h} V_i$ satisfying*

1. $w\left(E - \cup_{i=1}^{h} E\left(V_i\right)\right) = O\left(w(E)/\delta\right)$,
2. $\mathrm{R}_{\mathrm{diam}}(G\left[V_i\right]) = O\left(\delta^3 \cdot \frac{|V|}{w(E)}\right)$ *for all $i = 1, ..., h$.*

Making a constant δ large enough gives us the following corollary:

▶ **Corollary 2.17.** *There exists an absolute constant α such that given a weighted graph $G = (V, E, w)$, there is an algorithm with time complexity $\tilde{O}\left(m \cdot n \cdot \log\left(\frac{w(E)}{\min_e w(e)}\right)\right)$ that finds a partition $V = \bigcup_{i=1}^{h} V_i$ satisfying:*

1. $w\left(E - \cup_{i=1}^{h} E\left(V_i\right)\right) \leq w(E)/2$
2. $\mathrm{R}_{\mathrm{diam}}(G\left[V_i\right]) \leq \alpha \frac{|V|}{w(E)}$ *for all $i = 1, ..., h$.*

2.4 Spectral Sparsification via Bounded Independence Sampling

Doron et al. [11] give an algorithm for spectral sparsification via bounded independence sampling:

Algorithm 2.2 Sparsify$(G = (V, E, w), \{\tilde{R}_{ab}\}_{(a,b)\in E}, k, \varepsilon, \delta)$.

1. Initialize H to be the empty graph on $n = |V(G)|$ vertices.
2. Set $s \leftarrow \frac{18e \log n}{\varepsilon^2} \cdot \left(\frac{n}{\delta}\right)^{2/k}$.
3. For every edge $(a, b) \in E$, set $p_{ab} \leftarrow \min\left\{1, w_{ab} \cdot \widetilde{R}_{ab} \cdot s\right\}$
4. For every edge $(a, b) \in E$, add (a, b) to H with weight w_{ab}/p_{ab} with probability p_{ab}. Do this sampling in a k-wise independent manner.
5. Return H.

▶ **Theorem 2.18** ([11] Theorem 3.1). *Let $G = (V, E, w)$ be an undirected connected weighted graph on n vertices with Laplacian $\mathbf{L}_G$ and effective resistances $R = \{R_{ab}\}_{(a,b)\in E}$. Let $0 < \varepsilon < 1$, $0 < \delta < 1/2$, and let $k \le \log n$ be an even integer. Let H be an output of* Sparsify$(G, R, k, \varepsilon, \delta)$ *and let $\mathbf{L}_H$ be its Laplacian. Then, with probability at least $1 - 2\delta$ we have:*

1. $\mathbf{L}_H \cong_\varepsilon \mathbf{L}_G$,
2. *H has $O\left(\frac{1}{\delta^{1+2/k}} \cdot \frac{\log n}{e^2} \cdot n^{1+\frac{2}{k}}\right)$ edges.*

Note that if G is connected and $\mathbf{L}_H \cong_\varepsilon \mathbf{L}_G$ for some $\varepsilon \ge 0$, then H must also be connected. Thus, setting $\varepsilon = 1/2$, $k = \Theta(\log n)$, and $\delta = 1/\text{poly}(n)$, get that $s = \Theta(\log n)$ and $p_{ab} = \Theta(w_{ab} \cdot R_{ab} \cdot \log n)$, giving us the following corollary:

▶ **Corollary 2.19.** *Let $G = (V, E, w)$ be an undirected connected weighted graph on n vertices with effective resistances $\{R_{ab}\}_{(a,b)\in E}$. Then, for any $\delta = 1/\text{poly}(n)$, sampling edges of G in a $\Theta(\log n)$-wise independent manner with marginals $\Theta(w_{ab} \cdot R_{ab} \cdot \log n)$ for each $(a, b) \in E$ yields a connected graph H with probability $1 - 2\delta$.*

3 Connectedness Under Bounded-Independence Edge Subsampling

3.1 Leverage Scores and Minimal Cut Duality

In this section, we prove the following theorem:

▶ **Theorem 3.1.** *Given an unweighted graph $G = (V, E)$ with m edges and minimum cut c, there exists a weighting $w : E \to \mathbb{R}$ such that the weighted graph $G' = (V, E, w)$ satisfies the following:*

- *for each $e \in E$, the leverage score is $\ell(e) \le 4\alpha/c$, where α is an absolute constant from Corollary 2.17,*
- $w_{\max}/w_{\min} = m^{O(\log m)}$.

Moreover, the converse is also true: if there exists a weighting such that all leverage scores are at most $1/c$, the minimal cut of the original graph is $\ge c$.

We prove the theorem by recursively decomposing our graph into components with high effective resistance diameter, reweighting the edges within each component, and then contracting the graph on these components. We start by establishing the following useful lemma:

▶ **Lemma 3.2.** *Given a graph $G = (V, E, w)$ of effective resistance diameter R, and two disjoint vertex sets $s_1, s_2, ..., s_k$ and $t_1, t_2, ..., t_\ell$, let $\mathbf{f}$ be the unique electric flow induced by injecting current α_i to s_i and extracting β_j from t_j so that $\sum_i \alpha_i = \sum_j \beta_j = 1$. Then $\mathcal{E}(\mathbf{f}) \leq R$.*

Proof. Let $\mathbf{f}_{ij}$ be a unit electric flow from s_i to t_j. Then:

$$\mathcal{E}(\mathbf{f}_{ij}) = \mathrm{Reff}(s_i, t_j) \leq R$$

Now, consider the distribution of pairs (I, J) such that $P[I = i, J = j] = \alpha_i \beta_j$. If we pick the source s_I and the sink s_J according to this distribution, and induce a current as above, we will get some flow $\mathbf{f}_{IJ}$, which in expectation will be:

$$\mathbb{E}\left[\mathbf{f}_{IJ}\right] = \sum_{i,j} \alpha_i \beta_j \mathbf{f}_{ij} = \tilde{\mathbf{f}}$$

Importantly, $\tilde{\mathbf{f}}$ is not necessarily an electric flow, since it might not obey Ohm's law. However, the energy of $\tilde{\mathbf{f}}$ bounds the energy of the true electric flow with the same external currents.

Finally, note that by definition, $\mathcal{E}(\mathbf{f}) = \sum_e \mathbf{f}(e)^2 / w_e = \mathbf{f}^T \mathbf{R} \mathbf{f}$, where $\mathbf{R}$ is the diagonal matrix of resistances. Since $\mathbf{R}$ is positive semi-definite, then $\mathcal{E} : \mathbb{R}^n \to \mathbb{R}, \mathbf{f} \mapsto \mathcal{E}(\mathbf{f})$ is convex, meaning that by Jensen's inequality:

$$\mathcal{E}(\tilde{\mathbf{f}}) \leq \mathbb{E}_{(I,J)}\left[\mathcal{E}(\mathbf{f}_{IJ})\right] = \sum_{i,j} \alpha_i \beta_j \mathcal{E}(\mathbf{f}_{ij}) \leq R \sum_{i,j} \alpha_i \beta_j = R$$

Moreover, note that the external currents of $\tilde{\mathbf{f}}$ match exactly those of $\mathbf{f}$: the current injected at s_i is $\sum_j \alpha_i \beta_j = \alpha_i$ and the current extracted from t_j is $\sum_i \alpha_i \beta_j = \beta_j$. Since the actual electrical flow minimizes the energy, then the energy of $\mathbf{f}$ is $\mathcal{E}(\mathbf{f}) \leq \mathcal{E}(\tilde{\mathbf{f}}) \leq R$. ◀

We use this lemma to show that when a graph is contracted on some subgraphs, the effective resistance diameter of the contracted graph can be combined with effective resistance diameters of contracted components to bound the diameter of the original graph.

▶ **Lemma 3.3.** *Given a graph $G = (V, E, w)$ and its partitioning $V = \sqcup_{i=1}^{h} V_i$, let the contracted graph $G' = (V', E', w')$ be constructed by identifying vertices in the same partition set. Assume $\mathrm{R}_{\mathrm{diam}}(G') \leq R_1$ and $\mathrm{R}_{\mathrm{diam}}(G[V_i]) \leq R_0$ for all i. Then $\mathrm{R}_{\mathrm{diam}}(G) \leq R_1 + hR_0$.*

Proof. Consider any $s, t \in V$. We will show that $\mathrm{Reff}(s, t) \leq R_1 + hR_0$. Note that if $s, t \in V_i$ for some i, then the effective resistance between s and t in G is bounded by the effective resistance in $G[V_i]$, meaning that $\mathrm{Reff}(s, t) \leq R_0 \leq R_1 + hR_0$. Hence, assume $s \in V_i$, $t \in V_j$ for some $i \neq j$.

Note that the effective resistance equals the minimal energy of a unit flow passing from s to t. Hence, it is sufficient to construct some unit flow (not necessarily an electric flow) $\mathbf{f}$ such that $\mathcal{E}(\mathbf{f}) \leq R_1 + hR_0$.

Let $s', t' \in V'$ be the vertices of G' that correspond to contracted components V_i, V_j in G. Consider an electric unit flow $\mathbf{f}'$ from s' to t' in G': since $\mathrm{Reff}_{G'}(s', t') \leq \mathrm{R}_{\mathrm{diam}}(G') \leq R_1$, then $\mathcal{E}_{G'}(\mathbf{f}') = \sum_{e \in E'} \mathbf{f}'(e)^2 / w'(e) \leq R_1$. It remains to lift this flow to a flow in G.

Our lifted flow $\mathbf{f}$ would match $\mathbf{f}'$ on the flows of all edges crossing between partitions: namely, for each edge $e' \in E'$ of G', if the corresponding edge in G is $e \in E$, then we would force $\mathbf{f}(e) = \mathbf{f}'(e')$. Then, consider any subgraph $G[V_i]$ of G. The currents across the edges going between partitions force particular external flows to $G[V_i]$: specifically, if we limit our view to $G[V_i]$, any vertex $v \in V_i$ has external flow equaling the sum of all flows going to v

through these crossing edges (with the only exception of vertices s and t, where the external currents are 1 and -1, respectively). These external currents force a unique electric flow $\mathbf{f}_i$ through $G[V_i]$. Since the total external current flowing in $G[V_i]$ equals the total current flowing out of $G[V_i]$ and is bounded by 1 (the total current injected in G), then by Lemma 3.2 we get that the energy of this flow is $\mathcal{E}_{G[V_i]}(\mathbf{f}_i) \leq \mathrm{R}_{\mathrm{diam}}(G[V_i]) \leq R_0$.

Hence, we get the flow $\mathbf{f}'$ fixing the flow across the edges going in between partitions, and in each subgraph $G[V_i]$ we get a flow $\mathbf{f}_i$ consistent with the currents external to $G[V_i]$. Combining these flows, we get a unit s-t flow $\mathbf{f}$ in the original graph G. Finally, since the energy of the flow only depends on the flow across each edge and edge weights, we can safely combine the eneries of each of these flows to get the energy of $\mathbf{f}$:

$$\mathcal{E}(\mathbf{f}) = \mathcal{E}(\mathbf{f}') + \sum_{i=1}^{h} \mathcal{E}(\mathbf{f}_i) \leq R_1 + hR_0$$

Finally, since the effective resistance between s and t in G equals the minimal energy of a unit flow from s to t, the above expression bounds it from above: $\mathrm{Reff}_G(s,t) \leq \mathcal{E}(\mathbf{f}) \leq R_1 + hR_0$. ◀

With this lemma, we are finally ready to prove Theorem 3.1:

Proof of Theorem 3.1. The converse is fairly straightforward to see. Consider the weighting for which the leverage scores are bounded by $1/c$ and take any cut of G that has k edges. Then sample a spanning tree of G' with the probabilities proportional to the product of the edge weights in the tree. It is known that the probability of any edge being included in the spanning tree equals its leverage score, meaning that each cut edge will be included with probability at most $1/c$. Then, by the union bound, the probability that at least one cut edge is included is at most k/c. But one edge of any cut is definitely included in any spanning tree, meaning that $k/c \geq 1$, thus giving that any cut has size k is $\geq c$.

Now let us prove the main direction. Let $\Delta \in \mathbb{R}$ be the parameter to be chosen later. We will reweight the graph using Algorithm 3.1.

Algorithm 3.1 GraphReweight.

1. Set $i = 0$ and $G_0 = G$
2. On recursive step i, find a partition of $G_i = (V_i, E_i)$ into subgraphs $V_i = \sqcup_{j=1}^{h} V_{ij}$ according to Corollary 2.17, meaning that the number of edges going between partition sets is $\leq |E_i|/2$, and effective resistance diameter of $G_i[V_{ij}]$ is $\leq \alpha \cdot |V_i|/|E_i|$ for all j (where α is a constant from Corollary 2.17).
3. For each edge e in each $G_i[V_{ij}]$, set $w(e) = 1/\Delta^i$.
4. Contract each V_i in a single vertex, obtaining a multigraph $G_{i+1} = (V_{i+1}, E_{i+1})$, where $E_{i+1} = \{(u,v) \in E_i : u \in V_{ij}, v \in V_{ik} \text{ for } j \neq k\}$
5. If $|V_{i+1}| > 1$, return to step 2 for $i+1$, otherwise output w.

Consider any step i of the algorithm and any subgraph $G_i[V_{ij}]$ of G_i that would be contracted on this step. By a slight abuse of notation, write $G[V_{ij}]$ to be the subgraph of the original graph G induced by all the vertices in V that are contracted to vertices V_{ij} on steps 0 to $i-1$.

We will prove by induction that on each step i, for each subgraph $G[V_{ij}]$ of the original graph, the effective resistance diameter (with respects to the weights w assigned) is:

$$\mathrm{R}_{\mathrm{diam}}\left(G[V_{ij}]\right) \leq 2\alpha \cdot \Delta^i \cdot \left(1 + \frac{n}{\Delta}\right)^i \cdot \frac{1}{c},$$

where α is a constant from Corollary 2.17.

Note that since the minimal cut of G is c, then each vertex has degree at least c, meaning that the total number of vertices in G is at least $c|V|/2$. Then $|V|/|E| \leq |V|/(c|V|/2) = 2/c$, so by our choice of partitioning on step 2:

$$\mathrm{R}_{\mathrm{diam}}\left(G[V_{0j}]\right) = \mathrm{R}_{\mathrm{diam}}\left(G_0[V_{0j}]\right) \leq \alpha\frac{|V|}{|E|} = 2\alpha \cdot \frac{1}{c},$$

proving the base case of induction.

Then, consider any step $i \geq 1$ and any subgraph $G_i[V_{ij}]$ of G_i. Notice that the minimal cut of G_i is at least c, since contracting on subgraphs only limits the set of all cuts to those which don't cut through contracted subgraphs, meaning that it does not decrease the min cut. Hence, $|V_i|/|E_i| \leq 2/c$, so the effective resistance diameter of a subgraph found on Step 2 of Algorithm 3.1 is $\leq 2\alpha/c$. However, note that when we set the weights of all edges of $G_i[V_{ij}]$ to $1/\Delta^i$, the effective resistance diameter given above (which is the diameter for unit weights) will be scaled by Δ^i (since the resistance of each edge is scaled by Δ^i), becoming:

$$\mathrm{R}_{\mathrm{diam}}(G_i[V_{ij}]) \leq 2\alpha \cdot \Delta^i \cdot \frac{1}{c}$$

Then, consider any vertex $v_k \in V_{ij} \subseteq V_i$. This vertex was created by contracting some component $V_{(i-1)k}$ on the previous recursive step of the algorithm, so by the inductive hypothesis we get that:

$$\mathrm{R}_{\mathrm{diam}}\left(G[V_{(i-1)k}]\right) \leq 2\alpha \cdot \Delta^{i-1} \cdot \left(1 + \frac{n}{\Delta}\right)^{i-1} \cdot \frac{1}{c}$$

Note that $G_i[V_{ij}]$ of effective resistance diameter $2\alpha\Delta^i/c$ is obtained from $G[V_{ij}]$ by contracting at most n such subgraphs $G[V_{(i-1)k}]$ of effective resistance diameter $2\alpha \cdot \Delta^{i-1} \cdot \left(1 + \frac{n}{\Delta}\right)^{i-1} \cdot \frac{1}{c}$ each, meaning that by applying Lemma 3.3 we can bound $\mathrm{R}_{\mathrm{diam}}(G[V_{ij}])$ as:

$$\mathrm{R}_{\mathrm{diam}}(G[V_{ij}]) \leq 2\alpha\Delta^i \cdot \frac{1}{c} + 2\alpha n \cdot \Delta^{i-1} \cdot \left(1 + \frac{n}{\Delta}\right)^{i-1} \cdot \frac{1}{c} \leq 2\alpha \cdot \Delta^i \cdot \left(1 + \frac{n}{\Delta}\right)^{i} \cdot \frac{1}{c},$$

which concludes our inductive proof for $\mathrm{R}_{\mathrm{diam}}$.

Finally, consider any edge that was contracted on step i: $e \in G_i[V_{ij}]$. Then, clearly the effective resistance of e in G is upper bounded by the effective resistance of e in a subgraph of G, $G[V_{ij}]$, which in turn is upper bounded by $\mathrm{R}_{\mathrm{diam}}(G[V_{ij}])$. Then, since $w(e) = 1/\Delta^i$, we get that the leverage score of e in G is upper bounded by:

$$\ell_G(e) = w(e)\,\mathrm{Reff}_G(e) \leq w(e)\,\mathrm{Reff}_{G[V_{ij}]}(e) \leq \frac{\mathrm{R}_{\mathrm{diam}}(G[V_{ij}])}{\Delta^i} \leq 2\alpha \cdot \left(1 + \frac{n}{\Delta}\right)^{i} \cdot \frac{1}{c}$$

To conclude the proof, we observe that on each recursive step $|E_i|$ is reduced by a factor of 2, meaning that the total number of steps the algorithm takes to complete is $\log m$. Then, picking $\Delta = \Theta(n \log m)$ is sufficient to ensure that $\left(1 + \frac{n}{\Delta}\right)^{i} \leq \left(1 + \frac{n}{\Delta}\right)^{\log m} \leq 2$, resulting in all leverage scores being $\leq 4\alpha/c$.

For such Δ, get that $w_{\max}/w_{\min} = \Delta^{O(\log m)} = m^{O(\log m)}$. ◀

3.2 Concluding Connectedness Under Subsampling

Finally, we show that Theorem 3.1 can be combined with Theorem 2.18 to show that a graph with a large minimum cut stays connected even when edges are subsampled in a k-wise independent manner:

▶ **Theorem 3.4.** *Given an undirected unweighted graph $G = (V, E)$ on m edges, if the minimum cut of G is $c \log m$, then sampling the edges of G in a $O(\log m)$-wise independent manner with marginals $O(1/c)$ yields a connected graph with probability $1 - 1/\text{poly}(m)$.*

Proof. Theorem 3.1 tells us that there is a weighting w of G such that for each $(a, b) \in E$, the leverage score is $w_{ab} \cdot \tilde{R}_{ab} \leq 4\alpha/c \log m$. Then, by Corollary 2.19, $O(\log m)$-wise independent sampling with marginals $p_{ab} = \Theta(w_{ab} \cdot \tilde{R}_{ab} \cdot \log n) = \Theta(1/c)$ gives a connected graph with probability $1 - 1/\text{poly}(m)$. ◀

We also have the following immediate corollary:

▶ **Corollary 3.5.** *Given an undirected unweighted graph $G = (V, E)$ on m edges, there is a choice of constant κ such that if the minimum cut of G is $\kappa \log m$, then sampling the edges of G in a $O(\log m)$-wise independent manner with marginals $1/2$ yields a connected graph with probability $1 - 1/\text{poly}(m)$.*

In the full version, we show that these results can be extended to show that by choosing the marginal probabilities correctly, we can ensure that there is a non-negligible probability that *any* near-minimum cut is the *unique* surviving cut in our sample. Additionally, we construct a different distribution that satisfies the same connectedness results while needing only $O(\log m \cdot \log \log m)$ random bits.

4 Cycle-Freeness Under Almost Bounded-Independence Sampling

In this section, we consider using (almost) bounded-independence when sampling graphs with large girth. For these graphs, we show that even this bounded-independence sampling yields subsampled graphs with *no cycles* with high probability.

4.1 Cycle-Freeness Under High Girth

To start, we recall the work of Karp, Upfal, and Wigderson [19], who showed the following claim that characterizes when cycles appear under *uniform sampling* of edges:

▷ Claim 4.1. Let $G = (V, E)$ be a graph with m edges and shortest cycle length $> 20 \log(m)$. A uniformly random subsample of E at rate $1/2$ will (with high probability):

1. Contain at least $m/4$ edges.
2. Not have any cycles.

We include the proof here for completeness:

Proof. To start, observe that because the shortest cycle is of length $> 20 \log(m)$, this means that for any pair of vertices $i, j \in V$, there can be at most one path of length $10 \log(m)$ between i, j.

Now, let us consider sampling E at rate $1/2$ to yield $\hat{E}$. If $\hat{E}$ contains a cycle, then it must be a cycle of length $> 20 \log(m)$, as E itself had no cycles of shorter length, and $\hat{E} \subseteq E$. At the same time, if $\hat{E}$ contains a cycle C of length $> 20 \log(m)$, C must contain some path of length $10 \log(m)$. Thus, in order to show that $\hat{E}$ does not contain any cycles, it suffices to show that $\hat{E}$ does not contain any paths of length $10 \log(m)$. To see this, we recall that there is at most one path of such length for each pair of $i, j \in V$, and thus there are at most

$$\binom{|V|}{2} \leq \binom{2|E|}{2} \leq 2m^2$$

such paths. Because each path survives with probability $\leq \left(\frac{1}{2}\right)^{10\log(m)}$, we see that there is no such path with probability $\geq 1 - \frac{2m^2}{m^{10}} \geq 1 - \frac{2}{m^8}$. A simple Chernoff bound yields that the number of edges sampled is $\geq m/4$ with exponentially high probability, and thus conditions (1) and (2) are satisfied with probability $\geq 1 - \frac{1}{m^7}$. ◁

Now, we show how to use δ-almost k-wise independent distributions to derandomize the above sampling procedure:

▷ Claim 4.2. Let $G = (V, E)$ be a graph with m edges and shortest cycle length $> 20\log(m)$. Consider sampling the edges of G using a $(1/m^{100})$-almost $10\log(m)$-wise independent distribution with marginals $1/2$. Then, there is a sample from the distribution which:

1. Contains at least $m/10$ edges.
2. Does not have any cycles.

Proof. As in the proof of Claim 4.1, it suffices to prove that none of the $\leq 2m^2$ paths of length $10\log(m)$ survive the sampling. For each such path, let us denote the constituent edges by S. Because our sample space is δ-almost $10\log(m)$-wise independent, we know that over a random sample $X = (x_1, \ldots x_m)$ from our space,

$$d_{\mathrm{TV}}(X(S), U(S)) \leq \frac{1}{m^{100}}.$$

In particular, this means that

$$\Pr[X(S) \neq 1^S] \geq \Pr[U(S) \neq 1^S] - \frac{1}{m^{100}} = 1 - \frac{1}{m^{10}} - \frac{1}{m^{100}} \geq 1 - \frac{2}{m^{10}},$$

which means that

$$\Pr[\text{path } S \text{ survives}] \leq \frac{2}{m^{10}}.$$

Taking a union bound over all $\leq 2m^2$ paths then yields that with probability $\geq 1 - \frac{4}{m^8}$, no path of length $10\log(m)$ survives, and thus that no cycle survives the sampling.

Now, because we use a δ-almost $10\log(m)$-wise independent distribution with marginals $1/2$, we know that $\mathbb{E}[\sum_{i=1}^m X_i] = m/2$. By a simple Markov bound, we know then that $\Pr[\sum_{i=1}^m X_i \geq m/10] \geq 0.4$. So, by a union bound, we know that with probability $\geq 0.4 - \frac{4}{m^8}$, a sample from our distribution will have at least $m/10$ edges and not have any cycles. Because this probability is > 0, this yields the stated claim. ◁

We also can extend the result above to graphs with smaller shortest cycle length provided that we decrease the marginals accordingly:

▶ **Corollary 4.3.** *Given an undirected unweighted graph $G = (V, E)$ on m edges, if the shortest cycle length of G is $\ell \leq 20\log(m)$, then sampling the edges of G using Algorithm 2.1 with $p = (1/2)^{\lceil 40\log(m)/\ell \rceil}$ and $(1/m^{200})$-almost ℓ-wise independence yields a connected graph with probability $1 - 1/\mathrm{poly}(m)$.*

Proof. We construct an auxiliary graph G', whereby we replace each edge in G with a path of length $s = \lceil 40\log(m)/\ell \rceil$ (adding $s - 1$ new vertices for each edge). Immediately, we can see that the shortest cycle in G' is of length $\ell' \geq 40\log(m)$. At the same time, the number of edges in G' is $m' = sm < m^2$, since $s \leq m$ for sufficiently large m. Hence, we get that $\ell' \geq 40\log(m) > 20\log(m')$ and $\log(m') \leq 2\log(m)$, meaning that we can apply

Claim 4.2 to G'. Namely, if one samples the edges of G' using $(1/m^{2\cdot 100})$-almost $(2\cdot 10\log m)$-wise independent random bits with marginals $1/2$, the resulting graph is connected with probability $1 - 1/\mathrm{poly}(m)$.

Now, observe that our implementation of δ-almost ℓ-wise independent sampling with marginals $(1/2)^{\lceil 40\log(m)/\ell\rceil}$ from Algorithm 2.1 exactly corresponds to the above approach. ◀

4.2 Unique Cycle Survival Under Careful Sampling Rates

In this section, we extend the argument from the previous section to also address the setting of *cycles uniquely surviving* under sampling:

▷ Claim 4.4. Let $G = (V, E)$ be a graph with shortest cycle length ℓ, and let $C \subseteq E$ denote the edges participating in a cycle of length $\in [\ell, 1.01\ell]$. Now, let G/C denote the result of contracting the graph G on the edges C. Then, the shortest cycle length in G/C is $\geq 0.2\ell$.

Proof. Note a set of edges $C' \subseteq G/C$ forms a cycle in G/C if and only if $C' \cup C$ contains a cycle different from C in G. Thus, we instead focus on cycles $\hat{C} \subseteq G$, $\hat{C} \neq C$, and lower bound $|\hat{C} \setminus C|$.

It is easy to verify that for any pair of cycles $\hat{C} \neq C$, their symmetric difference $\hat{C} \oplus C = (\hat{C} \cup C) \setminus (\hat{C} \cap C)$ also contains a cycle (if we consider the edge-induced subgraph, any vertex in the symmetric difference would have an even degree).

Assume $|\hat{C} \setminus C| < 0.2\ell$. Then, since $|\hat{C}| \geq \ell$ (ℓ is the minimum cycle length), we have that $|\hat{C} \cup C| > 0.8\ell$. Then, since $|C| \leq 1.01\ell$, $|C \setminus C'| < 1.01\ell - 0.8\ell = 0.21\ell$. But then $|\hat{C} \oplus C| = |\hat{C} \setminus C| + |C \setminus \hat{C}| < 0.2\ell + 0.21\ell < \ell$. Since $\hat{C} \oplus C$ contains a cycle, and the minimum cycle length is ℓ, this is a contradiction. Hence, $|\hat{C} \setminus C| \geq 0.2\ell$, and so the shortest cycle length in G/C is $\geq 0.2\ell$. ◁

Now, we have the following claim:

▷ Claim 4.5. Let $G = (V, E)$ be a graph with shortest cycle length $\ell \leq 20\log(m)$, and let C be an arbitrary cycle in G of length $[\ell, 1.01\ell]$. Consider sampling the edges of G using a $(1/m^{500})$-almost 2ℓ-wise independent distribution, with marginals $p = (1/2)^{\lceil 200\log(m)/\ell\rceil}$. Then, with probability > 0, the resulting sample will:

1. Sample all edges involved in the cycle C.
2. Not contain any other cycle C'.

Proof. Because C is a cycle of length $[\ell, 1.01\ell]$, $\ell \leq 20\log(m)$, and $p = (1/2)^{\lceil 200\log(m)/\ell\rceil} \geq 1/(2m^{200/\ell})$, the probability C survives sampling is at least:

$$\Pr[C \text{ survives}] \geq \left(\frac{1}{2m^{200/\ell}}\right)^{1.01\ell} - \frac{1}{m^{500}} = \frac{1}{2^{1.01\ell}\cdot m^{202}} - \frac{1}{m^{500}} \geq \frac{1}{m^{250}},$$

where the last inequality follows since $2^{1.01\ell} \leq m^{20.2}$ for $\ell \leq 20\log(m)$.

At the same time, conditioned on C surviving, the only way for another cycle C' to survive is if the contracted graph G/C contains a cycle.

Consider the probability distribution of edges in G/C conditioned on C surviving sampling. Take any event A in this distribution (event depending on the edges in $E \setminus C$) that depends on $\leq 0.2\ell$ edges. Then:

$$\Pr[A | C \text{ survives}] = \frac{\Pr[A \wedge C \text{ survives}]}{\Pr[C \text{ survives}]} = \frac{\Pr_U[A \wedge C \text{ survives}] + e_1}{\Pr_U[C \text{ survives}] + e_2},$$

where $\Pr_U$ denotes the probabilities under fully independent sampling and e_1, e_2 are error terms. Since the event of C surviving depends on $|C| \leq 1.01\ell$ edges, then both of the events above depend only on $\leq 2\ell$ edges. Our (unconditional) distribution is δ-almost 2ℓ-wise independent, meaning that $|e_1|, |e_2| \leq \delta$. Moreover, for the independent distribution, $\Pr_U[A \wedge C \text{ survives}] = \Pr_U[A] \cdot \Pr_U[C \text{ survives}]$. Then note that:

$$\left|\Pr[A|C \text{ survives}] - \Pr_U[A]\right| = \left|\frac{\Pr_U[A] \cdot \Pr_U[C \text{ survives}] + e_1}{\Pr_U[C \text{ survives}] + e_2} - \Pr_U[A]\right|$$

$$\leq \frac{|e_1| + \Pr_U[A] \cdot |e_2|}{\Pr[C \text{ survives}]} \leq \frac{2\delta}{\Pr[C \text{ survives}]}$$

Hence, since $\delta = 1/m^{500}$ and $\Pr[C \text{ survives}] \geq 1/m^{250}$, we get that the distribution of edges in G/C conditioned on C surviving is $(1/m^{200})$-almost 0.2ℓ-wise independent.

Since the shortest cycle length in G/C is at least $\ell/5$ (as per Claim 4.4), by Corollary 4.3 we know that subsampling the edges of G/C using Algorithm 2.1 with $(1/m^{200})$-almost 0.2ℓ-wise independence and marginals $(1/2)^{\lceil 40\log(m)/0.2\ell \rceil}$ yields a cycle-free graph with probability $1 - 1/\operatorname{poly}(m)$[1]. Therefore:

$$\Pr[C \text{ survives} \wedge \nexists C' : C' \text{ survives}] = \Pr[C \text{ survives}] \cdot \Pr[G/C \text{ is cycle-free } |C \text{ survives}] \geq$$

$$\geq \frac{1}{m^{250}} \cdot (1 - 1/\operatorname{poly}(m)) > 0.$$

This yields the claim. ◁

References

1 Leonard Adleman. Two theorems on random polynomial time. In *19th Annual Symposium on Foundations of Computer Science (sfcs 1978)*, pages 75–83. IEEE Computer Society, 1978. doi:10.1109/SFCS.1978.37.

2 Vedat Levi Alev, Nima Anari, Lap Chi Lau, and Shayan Oveis Gharan. Graph clustering using effective resistance. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 41:1–41:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ITCS.2018.41.

3 Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k-wise independence versus k-wise independence. *Inf. Process. Lett.*, 88(3):107–110, November 2003. doi:10.1016/S0020-0190(03)00359-4.

4 Noga Alon and Asaf Nussboim. K-wise independent random graphs. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 813–822. IEEE, 2008. doi:10.1109/FOCS.2008.61.

5 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for $(\Delta + 1)$ vertex coloring. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 767–786. SIAM, 2019. doi:10.1137/1.9781611975482.48.

[1] Note that the number of edges in G/C is actually $m - |C| < m$; however, by using m instead of $m - |C|$ we only decrease the marginal probabilities $(1/2)^{\lceil 40\log(m)/0.2\ell \rceil}$ and error term $\delta = 1/m^{500}$, meaning that cycle-freeness is also preserved with these parameters. We also note that for $\ell \leq 20\log(m)$, we have $0.2\ell \leq 20\log(m - 1.01\ell) \leq 20\log(m - |C|)$ for sufficiently large m, and so our invocation of Corollary 4.3 is valid.

6 Eric Balkanski, Aviad Rubinstein, and Yaron Singer. An optimal approximation for submodular maximization under a matroid constraint in the adaptive complexity model. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 66–77, 2019. doi:10.1145/3313276.3316304.

7 Nikhil Bansal, Rudi A Pendavingh, and Jorn G Van der Pol. On the number of matroids. *Combinatorica*, 35:253–277, 2015. doi:10.1007/S00493-014-3029-Z.

8 András A. Benczúr and David R. Karger. Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 47–55. ACM, 1996. doi:10.1145/237814.237827.

9 Joakim Blikstad, Sagnik Mukhopadhyay, Danupon Nanongkai, and Ta-Wei Tu. Fast algorithms via dynamic-oracle matroids. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1229–1242, 2023. doi:10.1145/3564246.3585219.

10 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In *Proceedings of the 13th International Conference on Approximation, and 14 the International Conference on Randomization, and Combinatorial Optimization: Algorithms and Techniques*, APPROX/RANDOM'10, pages 504–517, Berlin, Heidelberg, 2010. Springer-Verlag. doi:10.1007/978-3-642-15369-3_38.

11 Dean Doron, Jack Murtagh, Salil P. Vadhan, and David Zuckerman. Spectral sparsification via bounded-independence sampling. *Electron. Colloquium Comput. Complex.*, TR20, 2020. URL: https://api.semanticscholar.org/CorpusID:211506132.

12 Jack Edmonds. Matroid intersection. In *Annals of discrete Mathematics*, volume 4, pages 39–49. Elsevier, 1979.

13 Stephen A. Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-nc. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 754–763. ACM, 2016. doi:10.1145/2897518.2897564.

14 Wai Shing Fung, Ramesh Hariharan, Nicholas J. A. Harvey, and Debmalya Panigrahi. A general framework for graph sparsification. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 71–80. ACM, 2011. doi:10.1145/1993636.1993647.

15 Harold N Gabow and Robert E Tarjan. Efficient algorithms for a family of matroid intersection problems. *Journal of Algorithms*, 5(1):80–131, 1984. doi:10.1016/0196-6774(84)90042-7.

16 Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. On the implementation of huge random objects. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '03, page 68, USA, 2003. IEEE Computer Society.

17 Per M Jensen and Bernhard Korte. Complexity of matroid property algorithms. *SIAM Journal on Computing*, 11(1):184–190, 1982. doi:10.1137/0211014.

18 David R. Karger. Global min-cuts in rnc, and other ramifications of a simple min-cut algorithm. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 21–30. ACM/SIAM, 1993. URL: http://dl.acm.org/citation.cfm?id=313559.313605.

19 Richard M. Karp, Eli Upfal, and Avi Wigderson. The complexity of parallel computation on matroids. In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 541–550. IEEE Computer Society, 1985. doi:10.1109/SFCS.1985.57.

20 Richard M. Karp, Eli Upfal, and Avi Wigderson. The complexity of parallel search. *J. Comput. Syst. Sci.*, 36(2):225–253, 1988. doi:10.1016/0022-0000(88)90027-X.

21 Sanjeev Khanna, Aaron Putterman, and Junkai Song. On the parallel complexity of finding a matroid basis. *arXiv preprint, IEEE 66th Annual Symposium on Foundations of Computer Science, FOCS 2025, to appear*, 2025. arXiv:2507.08194,.

22 Sanjeev Khanna, Aaron Putterman, and Junkai Song. Optimal parallel basis finding in graphic and related matroids. *arXiv preprint arXiv:2511.04826*, 2025. doi:10.48550/arXiv.2511.04826.

23 Sanjeev Khanna, Aaron Putterman, and Junkai Song. An $\widetilde{O}(n^{3/7})$ round parallel algorithm for matroid bases. *arXiv preprint arXiv:2605.03979*, 2026.

24 Robert Kleinberg and Seth Matthew Weinberg. Matroid prophet inequalities. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 123–136, 2012.

25 Andrew McGregor. Graph stream algorithms: a survey. *ACM SIGMOD Record*, 43(1):9–20, 2014. doi:10.1145/2627692.2627694.

26 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 213–223. ACM, 1990. doi:10.1145/100216.100244.

27 Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011. doi:10.1137/080734029.

28 Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM J. Comput.*, 40(4):981–1025, 2011. doi:10.1137/08074489X.

29 Ashok Subramanian. A polynomial bound on the number of light cycles in an undirected graph. *Inf. Process. Lett.*, 53(4):173–176, 1995. doi:10.1016/0020-0190(94)00202-A.

30 Salil P Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.