How Large Is Your Graph?*

Varun Kanade[1], Frederik Mallmann-Trenn[2], and Victor Verdugo[3]

1 **Department of Computer Science, University of Oxford, Oxford, United Kingdom, and The Alan Turing Institute, London, United Kingdom**
2 **Département d'Informatique, École normale supérieure, PSL Research University, Paris, France, and Department of Computer Science, Simon Fraser University, Burnaby, Canada**
3 **Département d'Informatique, École normale supérieure, PSL Research University, Paris, France, and Departmento de Ingeniería Industrial, Universidad de Chile, Santiago, Chile**

Abstract

We consider the problem of estimating the graph size, where one is given only local access to the graph. We formally define a query model in which one starts with a *seed* node and is allowed to make queries about neighbours of nodes that have already been seen. In the case of undirected graphs, an estimator of Katzir *et al.* (2014) based on a sample from the stationary distribution π uses $O\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ queries; we prove that this is tight. In addition, we establish this as a lower bound even when the algorithm is allowed to crawl the graph arbitrarily; the results of Katzir *et al.* give an upper bound that is worse by a multiplicative factor $t_{\text{mix}}(1/n^4)$.

The picture becomes significantly different in the case of directed graphs. We show that without strong assumptions on the graph structure, the number of nodes cannot be predicted to within a constant multiplicative factor without using a number of queries that are at least linear in the number of nodes; in particular, rapid mixing and small diameter, properties that most real-world networks exhibit, do not suffice. The question of interest is whether any algorithm can beat breadth-first search. We introduce a new parameter, generalising the well-studied conductance, such that if a suitable bound on it exists and is known to the algorithm, the number of queries required is sublinear in the number of edges; we show that this is tight.

1998 ACM Subject Classification G.3 Probability and Statistics

Keywords and phrases Estimation, Random Walks, Social Networks

Digital Object Identifier 10.4230/LIPIcs.DISC.2017.34

1 Introduction

Networks contain a wealth of information and studying properties of networks may yield important insights. However, most networks of interest are very large and ordinary users may have rather restricted access to them. One of most basic questions about networks is the number of nodes contained in them. For example, the number of pages on the world wide web (WWW) is estimated to be just shy of 50 billion at the time of writing.[1] Facebook currently reports having about one and three quarter billion users;[2] Twitter reports having about 300 million active users. It is undesirable to rely on a small number of sources for such information. At times we might be interested in more specific graphs for which there is

* See [11] for the full version, `http://arxiv.org/abs/1702.03959`.
1 `www.worldwidewebsize.com`
2 Here, billion refers to one thousand million, not a million million.

no public information available at all. Is there a way to estimate the total number of nodes using rather limited access to these graphs?

Our model is motivated by the kind of access ordinary users may have to graphs of interest. For example, most social network companies provide some sort of an application programming interface (API). In the case of the world wide web, one option is to simply crawl. The graph query access models we use are formally defined in Section 2.1. For now, we mention four specific networks, each of which captures an important modelling aspect. First, Facebook is an undirected graph of friendships, and as long as privacy settings allow it, it is possible to request the number of friends for a given user and the identity of the friends. Second, the world wide web, which is a directed graph – it is possible to extract out-links on a given webpage; however, there is no obvious method to access all in-links. The third is what we refer to as the "fan" network – for a specific user it is possible to query who she is a fan of, however in terms of her fans only the number is revealed.[3] And finally, the fourth is the twitter network, which is directed, however both the followers and followees of a given user are accessible, as far as privacy settings allow. Obviously, the method to estimate the number of nodes may be rather different in each case. It is worth mentioning that the twitter network can essentially be treated as an undirected graph, however it still leaves open the possibility that differentiating in-links and out-links leads to better estimators.

While memory and computational requirements do act as constraints when dealing with large graphs, possibly the most important one is the rate limits set on queries made using the API. Even when crawling the web, there is the risk of simply being blocked if large volumes of requests are sent to the same server. Thus, the most scarce resource in this instance is the number of queries made about the graph. Our query model counts these costs strictly – essentially every time a new node is discovered, its degree is revealed at unit cost and a unit cost is incurred for every neighbour requested.

1.1 Related Work

There is a large body of literature on estimating statistical properties of graphs, reflecting the relevance of and interest in studying complex networks. Much of this work is in the more applied literature and in particular, we were unable to pin down a precisely defined query model. The work most closely related to ours is that of Katzir *et al.* [13]. They consider the setting where a random sample drawn from the stationary distribution of the random walk is available. If the random walk mixes rapidly and a suitable bound on the mixing time is known, this can be simulated in the models we consider. They show that $O\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ queries suffice, where π is the stationary distribution and d_{avg} is the average degree. We show that this is tight when given access to a sample from the stationary distribution. When only neighbour queries are allowed, we show that this bound is tight up to a multiplicative factor of the mixing time and other polylogarithmic factors. It is worth mentioning what this bound actually yields in graphs where the average degree is small, something common to most real world networks. It is always the case that $\|\pi\|_2^{-1} \leq \sqrt{n}$, where equality holds in the case of regular graphs. Thus, the number of queries required is significantly sublinear in the number of nodes. For graphs with power law degree distributions with parameter $\beta = 2$, Katzir *et al.* calculated that $\|\pi\|_2^{-1} = O(n^{1/4}\log(n))$. See Section 3.1 for further discussion.

In more recent work, Hardiman and Katzir give another estimator based on counting

[3] Although it is not obvious which of the extant networks have this property, there are close approximations such as Blogger. In any case, it is a very natural intermediate model between the second and the fourth.

shared neighbours [12]. They give a slightly better bound on the number of nodes sampled than the earlier work of Katzir *et al.* [13]. However, it is unclear that this can be implemented efficiently in the query model in our paper. Cooper *et al.* [5] obtain estimators for the number of edges, triangles and nodes. The estimators are based on random walks over the graph, but in particular, for estimating the number of nodes, the transition probabilities are not inversely proportional to the degree. Their estimator relies on counting the time required for the k^{th} return to a particular vertex. It seems unlikely that either their random walk or their estimator can be implemented in a query efficient manner. Brautbar and Kearns [3] show that estimating the size of cycle takes $\Omega(\sqrt{n})$ queries in the model where one has access two types of queries: one can query (i) outgoing edges and (ii) a node chosen uniformly at random. They also obtained bounds on the query complexity of the maximum degree and the clustering coefficient. Musco *et al.* [17] develop a distributed collision based approach to estimate the graph size and the average degree – in this model several random walks traverse the graph and count the number of collisions with other random walks. Dasgupta *et al.* [7] provide an estimator of the average degree that uses $O(\log(U)\log\log(U))$ samples, where U is an upper bound on the maximum degree. Somewhat surprisingly, this seems to be an easier task than estimating the number of nodes.

Cooper and Frieze [4] estimate the graph size in a dynamic setting where the graph grows over time and there exists an agent that is allowed to move through the network every time a new node arrives. They prove that this agent visits asymptotically a constant fraction of the vertices. If one has access to the complete neighbourhood of a node when it is visited, it is shown by Mihail *et al.* that the expected time in which a walk discovers a power law random graph is sublinear [16]. There has been some older theoretical work on estimating the size of graphs motivated primarily by search algorithms [14, 15, 18]. They consider trees and directed acyclic graphs, however, the model of graph access is unrelated and they do not present bounds on the estimates.

Related to the question of estimating graph properties is that of testing whether graphs have specific properties. Property testing has received much attention in the last two decades with properties of undirected graphs being one of the important areas of focus (see *e.g.,* [9]). Directed graphs have received somewhat less attention; there are two main query models, *unidirectional* where only out-neighbours may be queried, and *bidirectional* where both in and out-neighbours may be queried [1]. Bender and Ron showed that there exist properties such as strong connectivity, where the query complexity may be either $O(1)$ or $\Omega(\sqrt{n})$ depending on the model used, even when allowing two-sided error. More recently, Czumaj *et al.* have shown that if a property can be tested with constant query complexity in the bidirectional setting, it can be tested with sublinear query complexity in the unidirectional model [6]. Although, these query models are closely related to the ones in this paper, a crucial aspect exploited by most property testing algorithms is the ability to sample nodes uniformly at random, something that is not available in our setting.

A closely related line of work is that on estimating properties of distributions from samples; of most interest, in our case is the support size. This problem has a long history going back at least to Good and Turing [10]. We only mention a few recent relevant results here. Given access to uniformly sampled elements from a set, $O(\sqrt{n})$ samples suffice to derive a good estimate of the set size using the birthday problem [8]. However, it is not clear how to sample from the uniform distribution over the nodes of the graph in the query model we consider. Valiant and Valiant show that support size can be estimated to a good accuracy using $O(n/\log(n))$ samples for any distribution [19, 20]. However, their result requires that any element in the support has $\Omega(1/n)$ probability mass, something that is not true for

the stationary distribution of a random walk on a graph. (In the case of directed graphs this problem becomes even more severe, since some nodes may have exponentially small stationary probability mass.)

1.2 Our contributions

Firstly, our contribution is to express the problem of estimating graph properties formally. As discussed in the introduction, networks of interest vary significantly in terms of what access might be easily available to an ordinary user. Keeping in mind the examples of Facebook, the web, the fan-network and Twitter, we introduce different types of oracles that provide access to the graph.

The focus of this work is on estimating the number of nodes. For any $\varepsilon > 0$ and $\delta > 0$, we say that an algorithm (with access to a query oracle) provides an ε-accurate estimate of the number of nodes, if with probability at least $1 - \delta$ it outputs $\widehat{n}$, such that $|n - \widehat{n}| \leq \varepsilon n$. The main quantity to be optimised is the number of oracle queries, though all algorithms considered in this paper are also computationally highly efficient. Allowed queries are defined precisely in Section 2.1. Here, we point out that a unit cost must be paid for every disclosed neighbour as well as to know the degree of a node. All algorithms have access to the *identifier* of one *seed* node in the graph to begin with. Throughout we will assume that ε and δ are constants and the use of $O(\cdot)$ and $\Omega(\cdot)$ notation in this paper hides all dependence on ε and δ.

Undirected Graphs. Katzir *et al.* [13] implicitly assume the ability to sample from the stationary distribution. They show that in this setting $O\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ samples from the stationary distribution π suffice, where d_{avg} is the average degree. If the graph is connected and a suitable bound on the mixing time exists which is known to the algorithm, $O(t_{\text{mix}}(1/n^4))$[4] queries suffice to draw one node from a distribution that is close (up to inverse polynomial factors in variation distance) to the stationary distribution using only neighbour queries. This gives an upper bound of $O\left(t_{\text{mix}}(1/n^4) \cdot \left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)\right)$ queries with a neighbour query oracle (Corollary 2). In terms of lower bounds, we establish that

- (Theorem 5) Any algorithm with access to random samples from the *stationary distribution* π and outputs a 0.1-accurate estimate of the number of nodes with probability at least 0.99, requires $\Omega\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ samples.
- (Theorem 7) Any algorithm with access to neighbour queries and outputs a 0.1-accurate estimate of the number of nodes with probability at least 0.99, requires $\Omega\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ queries.

We remark that there is a gap between the upper bound and lower bound when considering an oracle with neighbour access. A question left open by our work is whether the multiplicative factor of $t_{\text{mix}}(1/n^4)$ is required, or whether a more efficient estimator can be designed.

Directed Graphs. The estimator of Katzir *et al.* is not applicable in the setting when graphs are directed, unless the query model allows in-neighbour queries as well as out-neighbour queries, in which case all results in the undirected setting hold. The reason for this is that even if one did receive a sample drawn from the stationary distribution, it is no longer the case that $\pi_v \propto \deg(v)$, a crucial property exploited by the estimator of Katzir *et al.*. In

[4] $t_{\text{mix}}(1/n^4)$ is the expected time it takes until the random walk is $1/n^4$-close to the stationary distribution w.r.t. to the total variation distance.

fact, unless strong assumptions are made on the relationship between the in-degree and the out-degree (*e.g.,* being Eulerian), no simple expression for π in terms of degrees exists.[5]

We provide constructions of graphs that demonstrate that low average degree, rapid mixing and small diameter are not sufficient to design algorithms to estimate the graph size with sublinear (in n) query complexity. In fact, we show that when only given access to a sample from the stationary distribution, a superpolynomially large sample may be required even for graphs with constant average degree, logarithmic diameter and rapid mixing. The reason for this is that in the case of directed graphs most of the stationary mass can be concentrated on a very small number of nodes.

In order to understand the query complexity when neighbour-queries are allowed, we define a new parameter called *general conductance*. Roughly speaking a graph has ε-general conductance ϕ_ε, if every set containing at most $(1-\varepsilon)n$ nodes has at least ϕ_ε fraction of directed edges going out. If a suitable bound on the value of ϕ_ε is known, we show that a simple edge-sampling algorithm outputs an estimate to within relative error ε while using $O(n/\phi_\varepsilon)$ queries (Theorem 10) and that this is almost tight, in the sense that any algorithm that outputs an estimate that is even slightly better than ε requires at least $\Omega(n/\phi_\varepsilon)$ queries (Theorem 12). The algorithm only requires access to out-neighbours, while the lower bound holds even with respect to an oracle that allows in-neighbour queries, which means that it also applies in the case of undirected graphs.

1.3 Discussion

In terms of improvements to our results, the most interesting question is whether any reasonable subclass of directed graphs are amenable to significantly improved query complexity (ideally sublinear) for the problem of estimating the number of nodes. The constructions in Section 4 show that having low average degree, small diameter and rapid mixing is not enough. For undirected graphs, it is an interesting question whether the extra factor of mixing time t_{mix} must be paid, when only neighbour queries are allowed. It is conceivable that an improved estimator that can handle correlated pairs of nodes can be designed, so as to not waste all but one query for every t_{mix} queries. Finally, it'd be interesting to study the question of estimating other properties of graphs, number of edges, number of triangles, *etc.* in this framework.

The model choices we made reflect the publicly available access to most extant networks; in particular, we were very stringent with accounting – every neighbour query counts as unit cost. Many APIs return the list of neighbours, although in chunks of a fixed size, *e.g.,* 100. It is hard to argue that 100 should be treated as constant in the context of social networks. Nevertheless, if we wanted a list of all followers of Barack Obama, this would still result in a very large number of queries. A natural extension to the query model in this paper is to allow the entire neighbourhood (possibly restricted to the out-neighbourhood in the case of directed graphs) to be revealed at unit cost. Estimators such as the one involving common neighbours of Hardiman and Katzir [12] can be implemented efficiently under such a model. Understanding the query complexity of estimation in these more powerful models is an interesting question.

[5] We mention that the distribution is closely related to the PageRank distribution. However, the PageRank random walk jumps to a uniformly random node in the graph with a small probability; this is done to avoid problems when the graph is not strongly connected.

2 Model and Preliminaries

Graphs. Graphs $G = (V, E)$ considered in this paper may be directed or undirected; typically we assume $|V| = n$ and $|E| = m$, though if there is scope for confusion we use $|V|$ or $|E|$ explicitly. For undirected graphs, for a node $v \in V$, we denote by $N(v)$ its *neighbourhood*, *i.e.,* $N(v) = \{w \mid \{v, w\} \in E\}$, and its degree by $\deg(v) := |N(v)|$. In the case of directed graphs, we denote $N^+(v) := \{w \mid (v, w) \in E\}$ its *out-neighbourhood* and by $\deg^+(v) := |N^+(v)|$ its *out-degree*. Similarly, $N^-(v) := \{u \mid (u, v) \in E\}$ denotes its *in-neighbourhood* and $\deg^-(v) := |N^-(v)|$ its *in-degree*. Furthermore, d_{avg} denotes the average degree, i.e, $\sum_{v \in V} \deg(v)/n$. Whenever there is scope for confusion, we use the notations $\deg_G(u)$, $N_G(v)$, $d_{\text{avg}}(G)$, *etc.* to emphasise that the terms are with respect to graph G.

Random walks in graphs. A discrete-time lazy random walk $(X_t)_{t \geq 0}$ on a graph $G = (V, E)$ is defined by a Markov chain with state space V and transition matrix $P = (p(u, v))_{u,v \in V}$ defined as follows: For every $u \in V$, $p(u, u) = 1/2$ (*Laziness*). In the undirected setting, for every $v \in N(u)$, $p(u, v) = 1/(2 \deg(u))$. In the directed setting, for every $v \in N^+(u)$, $p(u, v) = 1/(2 \deg^+(u))$. The transition probabilities can be expressed in matrix form as $P = (I + \mathcal{D}^{-1}A)/2$, where A is the adjacency matrix of G, $\mathcal{D}$ is the diagonal matrix of node degrees (only out-degrees if G is directed), and I is the identity. Let $p^t(u, \cdot)$ denote the distribution over nodes of a random walk at time step t with $X_0 = u$. For the most part, we will consider (strongly) connected graphs. Together with laziness, this ensures that the stationary distribution of the random walk, denoted by π, is unique and given by $\pi P = \pi$. In the undirected case, the form of the stationary distribution is particularly simple, $\pi(u) = \deg(u)/(2|E|)$; furthermore, the random walk is reversible, *i.e.,* $\pi(u)p(u, v) = \pi(v)p(v, u)$. As before, π_G is used to emphasise that the stationary distribution is respect to graph G.

Mixing time. To measure how far $p^t(u, \cdot)$ is from the stationary distribution we consider the *total variation distance*; for distributions μ, ν over sample space Ω the total variation distance is $\|\mu - \nu\|_{\mathsf{TV}} = \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)|$. The *mixing time* of the random walk is defined as $t_{\text{mix}} := \max_{u \in V} \min\{t \geq 1 \mid \|p^t(u, \cdot) - \pi\|_{\mathsf{TV}} \leq e^{-1}\}$. Although the choice of e^{-1} is arbitrary, it is known that after $t_{\text{mix}} \log(1/\varepsilon)$ steps, the total variation distance is at most ε.

2.1 Query Model

In this section, we formally define the query model that allows us to access the graph. We consider four different neighbour query oracles, $\mathcal{O}$, $\overrightarrow{\mathcal{O}}$, $\overrightarrow{\mathcal{O}}(1)$ and $\overrightarrow{\mathcal{O}}(2)$. We also assume that all oracles have graphs stored as adjacency lists. In the case of directed graphs, there are two adjacency lists for every vertex, one for in-neighbours and one for out-neighbours. No assumption is made regarding the order in which the adjacency lists are stored. In words, $\mathcal{O}$ captures access to undirected graphs such as social networks like Facebook, $\overrightarrow{\mathcal{O}}$ captures directed graphs such as the world wide web, where only out-edges are available, $\overrightarrow{\mathcal{O}}(1)$ captures directed graphs such as fan-networks, where the in-degree but not in-neighbours may be available, and $\overrightarrow{\mathcal{O}}(2)$ captures directed graphs such as Twitter where access is available to both in-edges and out-edges.

All oracles also make use of labelling functions[6] ; for some space L, a labelling function $\ell : V \to L$ used by an oracle is an injection. We allow ℓ to be defined dynamically. The labelling function we use throughout the paper is the *consecutive labelling function* defined as follows: The label set is $L = \mathbb{N}$. If S denotes all the vertices labelled by the oracle so far, for a new vertex $v \notin S$ picked by the oracle, the label is assigned as follows: if $S = \emptyset$, $\ell(v) = 1$, else $\ell(v) = \max\{\ell(u) \mid u \in S\} + 1$. For these neighbour query oracles, any algorithm can essentially make two types of queries, (i) init, and (ii) (l, i) for undirected graphs or (l, i, etype) for directed graphs, where etype is either in, out. We assume that oracles use a labelling function ℓ.

(i) init: The oracle initialises a set $S := \{v\}$, where v is chosen to be an arbitrary node in the graph. The different oracles respond as defined below.
- $\mathcal{O}$ responds with $(\ell(v), \deg(v))$ for some $v \in V$. $\overrightarrow{\mathcal{O}}$ responds with $(\ell(v), \deg^+(v))$ for some $v \in V$.
- $\overrightarrow{\mathcal{O}}(1)$ and $\overrightarrow{\mathcal{O}}(2)$ both respond with $(\ell(v), \deg^+(v), \deg^-(v))$.

(ii) (l, i) or (l, i, etype): $\mathcal{O}$ only responds to query of type (l, i) and the remaining to queries of the type (l, i, etype).
- For query (l, i), if there is $v \in S$ such that $\ell(v) = l$ and $i \leq \deg(v)$, then $\mathcal{O}$ returns $(\ell(u), \deg(u))$, where u is the i^{th} element in the adjacency list of v. The oracle updates $S \leftarrow S \cup \{u\}$. Otherwise it returns null.
- For query (l, i, out), if there is $v \in S$ such that $\ell(v) = l$ and $i \leq \deg^+(v)$, then $\overrightarrow{\mathcal{O}}$ returns $(\ell(u), \deg^+(u))$, while both $\overrightarrow{\mathcal{O}}(1)$ and $\overrightarrow{\mathcal{O}}(2)$ return $(\ell(u), \deg^+(u), \deg^-(u))$ where u is the i^{th} element on the out-neighbour adjacency list of v. The oracles all update $S \leftarrow S \cup \{u\}$. Otherwise, all oracles return null.
- For query (l, i, in), $\overrightarrow{\mathcal{O}}$ and $\overrightarrow{\mathcal{O}}(1)$ always return null. If there exists $v \in S$, such that $\ell(v) = l$ and $i \leq \deg^-(v)$, then $\overrightarrow{\mathcal{O}}(2)$ returns $(\ell(u), \deg^+(u), \deg^-(u))$, where u is the i^{th} element on the adjacency list of in-neighbours of v; otherwise, it returns null. In the case of $\overrightarrow{\mathcal{O}}(2)$, it updates $S \leftarrow S \cup \{u\}$.

It is worth pointing out that a response of null provides no new information to the algorithm. The algorithm only knows that it hadn't received l as a label before or that the degree of the node with label l is strictly smaller than i, things it already knew. This is because the oracle maintains a history of past queries; if this were not the case the algorithm could generate (random) labels and try to find out whether they corresponded to nodes in the graph. However, even for oracles that don't maintain such state explicitly, by choosing ℓ to be collision-resistant hash function, essentially the same behaviour can be achieved.

In some of our proofs, we also allow oracles to return side information. These are denoted by a superscript s, *e.g.,* $\mathcal{O}^s$. Access to oracles with (truthful) side information can only help to reduce query complexity, since an algorithm can choose to ignore any side information it receives. Finally, we say that an algorithm is *sensible* if it does not make a query to which it already knows the answer. It is clear that given any algorithm, there is a sensible algorithm that is at least as good as the original algorithm. The *sensible* algorithm merely simulates the original algorithm and whenever the original algorithm made a query to which the answer was known, the *sensible* algorithm simulates the oracle response. Finally, we consider the *stationary query oracle*, $\mathcal{O}^\pi$, which when queried returns $(\ell(v), \deg(v))$, where $v \sim \pi$; π is the stationary distribution.

[6] We do assume that there is unit cost in sending the label of a node to the algorithm, thus implicitly we may think of every label having a bit-representation that is logarithmic in the size of the graph. However, the bit-length of the label reveals minimal information, which we assume that the algorithm has access to anyway.

3 Undirected graphs

In this section, we focus on the question of estimating the number of nodes in undirected graphs. We show that the results obtained by Katzir *et al.* [13] are essentially tight, up to a factor of the mixing time and polylogarithmic terms in n. This suggests that rapid mixing is a critical condition for being able to estimate the size of the graph.

3.1 Results of Katzir *et al.*

In this section, we discuss the result of Katzir *et al.* [13] regarding estimating the number of nodes in a graph. Though, they don't discuss this formally, their method essentially boils down to having access to the stationary query oracle $\mathcal{O}^\pi$. For the sake of completeness, we outline the estimator of Katzir *et al.* here. Let

$$X = \{(\ell(x_1), \deg(x_1)), (\ell(x_2), \deg(x_2)), \ldots, (\ell(x_r), \deg(x_r))\}$$

be drawn from the stationary query oracle $\mathcal{O}^\pi$. Let $\Psi_1 = \sum_{i=1}^{r} \deg(x_i)$ and $\Psi_{-1} = \sum_{i=1}^{r} 1/\deg(x_i)$; let $C = \sum_{i \neq j} \mathbb{1}(\ell(x_i) = \ell(x_j))$ denote the random variable counting the number of collisions. Then, it is fairly straightforward to see that $\mathbb{E}[\Psi_1 \Psi_{-1}] = r + 2n\binom{r}{2} \|\pi\|_2^2$ and $\mathbb{E}[C] = 2\binom{r}{2} \|\pi\|_2^2$. Katzir *et al.* use the the following as an estimator for the number of nodes $\widehat{n} := \frac{\Psi_1 \Psi_{-1} - r}{C}$. They prove the following result.

▶ **Theorem 1** (Katzir et al. [13]). *Let $\varepsilon > 0$ and $\delta > 0$. Suppose that the number of samples is $r \geq 1 + \frac{32}{\varepsilon^2 \delta} \max\left\{\frac{1}{\|\pi\|_2}, d_{\text{avg}}\right\}$, where $d_{\text{avg}} = 2m/n$. Then, $\mathbb{P}[|\widehat{n} - n| \geq \varepsilon n] \leq \delta$.*

Given a bound T on t_{mix} and access to oracle $\mathcal{O}$, the stationary query oracle $\mathcal{O}^\pi$ can be approximately simulated. (We assume that the graph is connected.) We simply perform a random walk on the graph for $T \log(1/\rho)$ steps, if a sample from a distribution at most ρ far from π in total variation distance is desired. Using the above theorem, we get the following straightforward corollary. The proof of the corollary follows by simulating s random walks for $O(T \log s)$ time steps (queries) ensuring that each random walk has a total variation distance of s^{-2} from π.

▶ **Corollary 2.** *Let $\varepsilon > 0$ and $\delta > 0$. For a connected, undirected graph, $G = (V, E)$ let T be such that $t_{\text{mix}} \leq T$. Then there exists an algorithm that given T and access to oracle $\mathcal{O}$, outputs $\widehat{n}$, such that, $\mathbb{P}[|\widehat{n} - n| \geq \varepsilon n] \leq \delta$. Furthermore, the number of queries made by the algorithm to $\mathcal{O}$ is $O(Ts \log s)$, where $s = O\left(\frac{1}{\varepsilon^2 \delta} \cdot \max\left\{\frac{1}{\|\pi\|_2}, d_{\text{avg}}\right\}\right)$.*

The key question is, are these bounds tight? In Section 3.2, we give much stronger lower bounds, where we show that for any valid degree sequence $\mathbf{d} = (d_1, \ldots, d_n)$ on n nodes, there exists a sequence $\mathbf{d}'$ on $2n$ nodes, such that any algorithm with access to a stationary oracle for either sequence $\mathbf{d}$ or $\mathbf{d}'$, cannot distinguish between the two unless it makes $\Omega\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ queries. Here, we discuss simple instances where the bound in Theorem 1 is tight; we don't give formal proofs which are relatively straightforward.

- Let G_1 and G_2 be d-regular graphs on n and $2n$ nodes respectively, for $d < n$. Thus, samples from the stationary distribution are essentially just uniformly chosen random nodes in the graph. As the degrees are identical, they reveal no additional information. Thus, the algorithm has to query until a collision is observed, which requires $\Omega(\sqrt{n})$ queries. Clearly for these graphs $\frac{1}{\|\pi\|_2} = \Theta(\sqrt{n})$.

- We define the *sun* graph as a K_n with an additional edge out of each vertex to a node with degree 1. The *bright sun* graph is the same as the sun graph, except that there is a path of length 2 coming out of each vertex, rather than just an edge. Under the stationary oracle model, $\Omega(n)$ queries are required before any degree 1 or 2 nodes are returned since the total probability on these nodes is $O(1/n)$. As can be seen for these graphs $d_{\text{avg}} = \Theta(n)$.

The above examples show that there are graphs for which the bound of Katzir *et al.* is tight. In this paper, we show a significantly stronger statement – given any degree sequence $\mathbf{d}$, there are graphs for which the bound is almost tight.

3.2 Lower Bounds in the Stationary Query Model

In this section, we show that for any undirected, connected graph $G = (V, E)$, there exists an undirected, connected graph $\widetilde{G}$, such that any algorithm which has access to either the oracle $\mathcal{O}^{\pi}(G)$ or $\mathcal{O}^{\pi}(\widetilde{G})$, cannot distinguish between the two with significant probability without making $\Omega\left(\frac{1}{\|\pi\|_2} + d_{\text{avg}}\right)$ queries. Thus, it cannot output an estimate $\widehat{n}$ satisfying $|\widehat{n} - |V|| < |V|/2$ with probability 2/3. The graphs $(\widetilde{G})$ we construct are constructed as follows. For the lower bound of $\Omega\left(\frac{1}{\|\pi\|_2}\right)$ we consider two connected copies of the original graph. Using a coupling we show, by considering the total variation distance, that the resulting graph is indistinguishable from the original graph unless a node is sampled twice ("collision"). For the lower bound of $\Omega\left(d_{\text{avg}}\right)$ we augment the original graph by adding a 3-regular expander to it. This time we use a coupling together with the variation distance to show that it requires $\Omega(d_{\text{avg}})$ queries to sample at least once a node from the 3-regular expander and hence making the graphs indistinguishable if fewer samples are taken.

▶ **Lemma 3.** *Let $G = (V, E)$ be an undirected, connected graph with $|V| = n$ and $|E| \geq n$. Then there exists a graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$, with $|\widetilde{V}| = 2n$, such that any algorithm given access to either $\mathcal{O}^{\pi}(G)$ or $\mathcal{O}^{\pi}(\widetilde{G})$ with equal probability, cannot distinguish between the two with probability greater than $\frac{2}{3}$, unless it makes at least $\Omega\left(\frac{1}{\|\pi_G\|_2}\right)$ queries. As a consequence, no algorithm can output $\widehat{n}$ satisfying $|\widehat{n} - n^*| < n^*/2$ w.p. at least 2/3, where $n^* = n$ if the chosen graph is G and $2n$ if it is $\widetilde{G}$.*

▶ **Lemma 4.** *Let $G = (V, E)$ be an undirected, connected graph with $|V| = n$ and $|E| \geq n$. Then there exists a graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$, with $|\widetilde{V}| = 2n$, such that any algorithm given access to either $\mathcal{O}^{\pi}(G)$ or $\mathcal{O}^{\pi}(\widetilde{G})$ with equal probability, cannot distinguish between the two with probability greater than $\frac{2}{3}$, unless it makes at least $\Omega\left(d_{\text{avg}}(G)\right)$ queries. As a consequence, no algorithm can output $\widehat{n}$ satisfying $|\widehat{n} - n^*| < n^*/2$ w.p. at least 2/3, where $n^* = n$ if the oracle chosen corresponds to G and $n^* = 2n$ if it corresponds to $\widetilde{G}$.*

The proof can be found in the full version. As a consequence of Lemma 3 and Lemma 4, we get the following theorem.

▶ **Theorem 5.** *Given an undirected, connected graph $G = (V, E)$, there exist graphs $\widetilde{G}$, $\bar{G}$ with $2|V|$ nodes and a constant $p < 1$, such that any algorithm that is given access to one of three oracles $\mathcal{O}^{\pi}(G)$, $\mathcal{O}^{\pi}(\widetilde{G})$ and $\mathcal{O}^{\pi}(\bar{G})$, chosen with equal probability, requires $\Omega\left(\frac{1}{\|\pi_G\|_2} + d_{\text{avg}}(G)\right)$ queries to distinguish between them with probability at least p. As a consequence, any algorithm that outputs $\widehat{n}$, such that $|\widehat{n} - n^*| < n^*/2$ requires at least $\Omega\left(\frac{1}{\|\pi_G\|_2} + d_{\text{avg}}(G)\right)$ queries, where $n^* = n$ if the graph is G and $n^* = 2n$ if the graph is either $\widetilde{G}$ or $\bar{G}$.*

3.3 Oracle Sampling from the Neighbour Query Model

In this section, we show that with access to the oracle $\mathcal{O}(G)$, any algorithm that predicts the number of nodes in a graph G to within a small constant fraction requires $\Omega\left(\frac{1}{\|\pi_G\|_2} + d_{\text{avg}}\right)$ queries. For proving the lower bounds we use graphs generated according to the *configuration model* [2]. A vector $\mathbf{d} = (d_1, d_2, \ldots, d_n)$ is said to be *graphical* if there exists an undirected graph on n nodes such that vertex $i \in [n]$ has degree d_i. We briefly describe here how graphs are generated in the configuration model:

1. Create disjoint sets W_i, for $i \in \{1, \ldots n\}$, with $|W_i| = d_i$. The elements of W_i are called *stubs*.
2. Create a uniform random maximum matching in the set $\bigcup_{i=1}^{n} W_i$ (note that $\sum_{i=1}^{n} d_i$ must be even since $\mathbf{d}$ is graphical).
3. For a stub edge $\{x, y\}$ in the matching, such that $x \in W_i$ and $y \in W_j$, the edge $\{i, j\}$ is added to the graph.

The above procedure creates a graph where vertex i has degree exactly d_i. However, the graph may not be simple, *i.e.,* it may have multiple edges and self-loops. Also, this procedure does not necessarily produce a uniform distribution over graphs having degree sequence $\mathbf{d}$. The expected number of multi-edges and self-loops in the graph is, for many interesting graph families, only a small fraction and in any graph with bounded degree their expected number is a constant. We use $G \sim \mathcal{G}(\mathbf{d})$ to denote that a graph G was generated in the configuration model with degree sequence $\mathbf{d}$.

Recall the definition of a *sensible* algorithm as one that never makes a query to which it already knows the answer. A *sensible* algorithm has the following behaviour: (i) for every $u \in V$ and $i \leq \deg(u)$ ($\deg^+(u)$ and $\deg^-(u)$, respectively) it makes the query $(\ell(u), i)$ ($(\ell(u), i, \mathsf{in})$ and $(\ell(u), i, \mathsf{out})$, respectively) at most once, and (ii) it never queries a $(\ell(v), i)$ if it has not received $\ell(v)$ as a valid label or if $i \notin [1, \deg(v)]$. Note that there exists for any algorithm, in the above defined query model, a sensible implementation which needs at most as many queries as the original algorithm.[7] Clearly, any sensible algorithm wouldn't query $(\ell(u), j)$ after querying $(\ell(v), i)$. The degree sequences we uses to construct the lower bound graphs are essentially the same as in the stationary query model. This time, however, we are facing changing probability distributions which depend on the random choices revealed so far. The proof of the following lemma can be found in the full version.

▶ **Lemma 6.** *Let $\varepsilon > 0$. Let $\boldsymbol{d} = (d_1, \ldots, d_n)$ be an arbitrary graphical sequence and let $D := \sum_{i=1}^{n} d_i$. Let $G \sim \mathcal{G}(d)$, and let $\widetilde{G}$ be an arbitrary graph with degree sequence $\boldsymbol{d}$. There exists an implementation of an oracle $\mathcal{O}^s(G)$ (with side-information) such that if $(\ell(X_t), \deg_G(X_t))_{t=1}^{T}$ is the sequence of responses to a* sensible *algorithm, where T equals*

$$\min\left\{\min\{\tau \geq 0 \mid \textit{all neighbours of all known nodes are disclosed using } \tau \textit{ queries}\}, \frac{\varepsilon\sqrt{D}}{16}\right\},$$

and if $(\ell(\widetilde{X}_t), \deg_{\widetilde{G}}(\widetilde{X}_t))_{t=1}^{T}$ is the sequence returned by oracle $\mathcal{O}^\pi(\widetilde{G})$, then there exists a coupling so that the sequences $((\ell(X_t), \deg_G(X_t)))_{t=1}^{T}$ and $((\ell(\widetilde{X}_t), \deg_{\widetilde{G}}(\widetilde{X}_t)))_{t=1}^{T}$ are identical with probability at least $1 - \varepsilon$.

[7] For technical reasons, in the proof of Lemma 6, we use an oracle $\mathcal{O}^s$ with *s*ide-information as an extension of $\mathcal{O}$: $\mathcal{O}^s$ returns, upon query, exactly the same information as $\mathcal{O}$, but can add additional truthful information. In particular, we allow the oracle when queried $(\ell(v), i)$ to not only return the corresponding node $(\ell(u), \deg(u))$, where u is the i^{th} element in the adjacency list of v, but also the index, say j, in the adjacency list of u which corresponds to v.

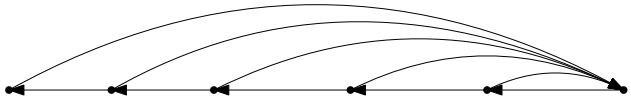

Figure 1 The Line graph on 6 nodes.

The proof of the main theorem of this section can be found in the full version.

▶ **Theorem 7.** *Let* $\boldsymbol{d} = (d_1, \ldots, d_n)$ *be a graphical vector satisfying* $\min_i d_i \geq 3$*. Then there exists a graphical* $\widetilde{\boldsymbol{d}} = (\widetilde{d}_1, \ldots, \widetilde{d}_n, \widetilde{d}_{n+1}, \ldots, \widetilde{d}_{2n})$ *with* $\min_i \widetilde{d}_i \geq 3$*, such that for* $G \sim \mathcal{G}(\boldsymbol{d})$ *and* $\widetilde{G} \sim \mathcal{G}(\widetilde{\boldsymbol{d}})$*, there exists* $c > 0$*, such that any algorithm with access to one of two oracles* $\mathcal{O}(G)$ *or* $\mathcal{O}(\widetilde{G})$ *chosen with equal probability, cannot distinguish between the two with probability greater than* $1 - c$ *unless it makes* $\Omega\left(\frac{1}{\|\pi_G\|_2} + d_{\text{avg}}(G)\right)$ *queries to the oracle.*

4 Directed Graphs

In this section, we consider the query complexity of estimating the number of nodes in directed graphs. We first observe that estimating n using the approach of Katzir *et al.* [13] is not possible since the stationary distribution of a node is in general not proportional to its degree. Another obstacle is that the stationary distribution of a node can be exponentially small as the graphs in Figure 1 and Figure 2 illustrate. In particular, it takes an exponentially large sample drawn from the stationary distribution to distinguish between the line graph of Figure 1 on n nodes and the line graph on $2n$ nodes, since the probability mass of the additional nodes is $2^{-\Omega(n)}$. It is also not very difficult to show that even with access to one of the two oracles $\overrightarrow{\mathcal{O}}$ or $\overrightarrow{\mathcal{O}}(1)$, $\Omega(n)$ queries are required to distinguish the line graph on n vertices from the line graph on $2n$ vertices.

As the example of the line graph reveals, unlike in the undirected case, rapid mixing and low average degree are not sufficient conditions to design a good estimator of the number of nodes using sublinear number of queries. The line graph shows that in the directed case, rapid mixing does not imply short (directed) diameter. One might hope that if one throws small diameter into the mix, in addition to low average degree and rapid mixing, a better estimator could be designed. Below, we show that this is not the case. The problem of estimation remains stubbornly hard, and $\Omega(n)$ queries to the oracle $\overrightarrow{\mathcal{O}}$, and $2^{\Omega(n)}$ queries to the stationary query oracle are required to achieve a good estimate of the number of nodes.

These observations suggest that exploring the graph, *e.g.,* through breadth-first search, is much faster than sampling from the stationary distribution. The question of interest is whether there is a property, satisfied by graphs of interest, which yields a query complexity better than $\Omega(m)$. We answer this positively in Section 4.1, where we introduce a parameter that generalises the conductance ϕ_ε and give almost tight bounds on the number of queries required to estimate n up to an ε relative error. Our Algorithm EdgeSampling takes this parameter as an input and terminates after $O(n/\phi_\varepsilon)$ queries which can be much smaller than the sample complexity of breadth-first search.[8]

[8] The results in Section 3 can be used if access to $\overrightarrow{\mathcal{O}}(2)$ is provided. In this case, we can simply treat the graph as being undirected. However, it is still interesting to understand whether the distinction between in-neighbours and out-neighbours allows one to design better estimators. At present, we are unaware of any graphs where this might be the case.

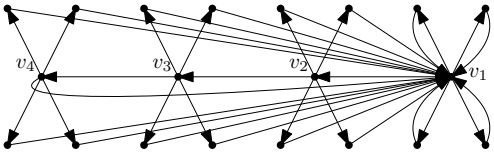

Figure 2 The graph Comet$(20, 4)$.

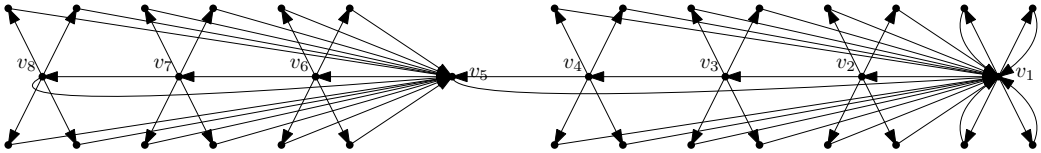

Figure 3 The graph DoubleComet$(40, 8)$. DoubleComet$(2n, 2k)$ consists of two copies of Comet(n, k) connected in the following way. First we remove all directed neighbours of v_{k+1} as well as the edge (v_k, v_1). Then we add (v_k, v_{k+1}), (v_{k+1}, v_{k+2}) as well as (v_{k+1}, v_1).

The Comet Graph

The *Comet graph*, Comet(n, k) is constructed as follows. Assume that k divides n. There is a directed cycle on the vertices $v_1, v_2, \ldots, v_k$, with edges (v_i, v_{i+1}) for $1 \leq i < k$ and (v_k, v_1). We denote these k vertices as *centres*. For every $\ell \in [k]$, there is a directed star $S_\ell = \{(v_\ell, v_{\ell,j}) : j \in [n/k - 1]\}$ with centre in v_ℓ of degree $n/k - 1$. For each leaf $v_{\ell,j}$ in star S_ℓ with $\ell \in [k], j \in [n/k - 1]$, there is a directed edge to the first star centre v_1, that is, $\{(v_{\ell,j}, v_1) : \ell \in [1, k], j \in [n/k - 1]\}$. We write v_ℓ^G and $v_{\ell,j}^G$ to emphasise that the nodes belong to graph G.

▶ **Theorem 8.** *Let n be a multiple of k. Then Comet(n, k) has mixing time $t_{\text{mix}} = O(1)$ and diameter k. Furthermore, any algorithm requires at least $\Omega((\frac{n}{k})^{k-1})$ queries to $\mathcal{O}^\pi$ and $\Omega(n)$ queries to $\overrightarrow{\mathcal{O}}$ to distinguish between $G =$ Comet(n, k) and $\widetilde{G} =$ Comet$(2n, 2k)$.*

The proof can be found in the full version. Note that the above results only apply to the oracles $\overrightarrow{\mathcal{O}}$ and $\mathcal{O}^\pi$, but not to $\overrightarrow{\mathcal{O}}(1)$, since the in-degrees make it easy to distinguish between the two graphs. However, it is straightforward to extend the graph such that the sample complexity remains $\Omega(n)$ even if the in-degrees are known; thus even with access to $\overrightarrow{\mathcal{O}}(1)$, $\Omega(n)$ queries are required.

▶ **Observation 9.** *Let n be a multiple of k. Then DoubleComet$(2n, 2k)$ (defined in Figure 3) has mixing time $t_{\text{mix}} = O(1)$ and diameter $2k$. Furthermore, any algorithm requires at least $\Omega(n)$ queries to $\overrightarrow{\mathcal{O}}(1)$ to distinguish between $G =$ Comet(n, k) and $\widetilde{G} =$ DoubleComet$(2n, 2k)$ on $2n$ nodes.*

4.1 Assuming a Bound on the Connectivity

In this section we introduce the parameter *general conductance*. We first recall some graph notation in the directed setting. Given a non-empty proper subset of vertices $S \subset V$, let $\deg^+(S) = |\{(u, v) \in E : u \in S\}|$ be the out-degree of S. The *cut* of S, ∂S, is the set of edges crossing between S and $V \setminus S$, that is, $\partial S = \{(u, v) \in E : u \in S, v \notin S\}$. The *general conductance* of S, $\phi(S)$, is the ratio between the cut of S, and the out-degree of S. That is, $\phi(S) = |\partial S| / \deg^+(S)$. Given $\varepsilon \in (0, 1)$, the graph ε-general conductance, ϕ_ε, is the minimum of $\phi(S)$ over every non-empty proper subset of V of size at most $(1 - \varepsilon)|V|$, *i.e.*,

$\phi_\varepsilon(G) = \min_{S\subseteq V: 1\leq|S|\leq(1-\varepsilon)|V|} \phi(S)$. Note that the parameter ϕ_ε decreases monotonically as ε decreases. In the undirected setting for $\varepsilon = 1/2$ this is just what is commonly known as the conductance.[9] In the following we describe the algorithm that estimates the graph size.

Upper bound in terms of the general conductance

We consider algorithm EdgeSampling for estimating the number of nodes. The algorithm takes as input the parameter ϕ, a lower bound on the general conductance ϕ_ε. The query complexity is $O(n/\phi_\varepsilon)$ and the output estimate $\widehat{n}$ satisfies $(1-\varepsilon)n \leq \widehat{n} \leq n$ with arbitrary confidence controlled by an input parameter ℓ. Observe that $O(n/\phi_\varepsilon)$ can be much smaller than the run time of breadth-first search $\Omega(m)$.

Algorithm overview. The algorithm works as follows. At each time step the algorithm maintains a counter Y. If at some point the counter exceeds the threshold ℓ, then the algorithm terminates. The algorithm divides the queries into blocks of length at most $2/\phi$ corresponding to the execution of the **for** loop. In each block, at every step the algorithm samples one outgoing edge uniformly at random from those available and not queried before. If at any step a new node is disclosed, then this finishes the block (break of the **for** loop) and the counter Y is decreased by 1. If the block finishes without finding a new node, then the counter is increased by 1. Once the counter reaches ℓ, which will happen eventually, then the algorithm outputs the number of nodes it discovered. Even though our goal is to minimise the query complexity, it is worth noticing that the time and space complexity can be kept low by choice of suitable data structures. Although the oracle returns labels of nodes, we use nodes and their labels interchangeably in the algorithm and the analysis of Theorem 10. The proof can be found in the full version.

▶ **Theorem 10.** *Algorithm* EdgeSampling$(\overrightarrow{\mathcal{O}}, \ell, \phi)$ *on graph G has a query complexity of* $\min\{2(2n+\ell)/\phi, m\}$ *and outputs an estimate $\widehat{n} \leq n$. Furthermore, if G has general conductance $\phi_\varepsilon(G)$ of at least ϕ, then the algorithm satisfies $\widehat{n} \geq (1-\varepsilon)n$ w.p. at least $1-2^{-\ell}$.*

▶ **Observation 11.** *The error made by Algorithm* EdgeSampling *is one-sided – the estimate never exceeds n. Allowing a two-sided error and given ε, one can instead output an estimate with smaller additive error. Consider a modification of Algorithm* EdgeSampling$(\overrightarrow{\mathcal{O}}, \ell, \phi)$ *on graph G which takes the additional parameter ε and outputs $\widehat{n}^* := |S_t|(1+\frac{\varepsilon}{2-\varepsilon})$ instead of $|S_t|$. If G has general conductance $\phi_\varepsilon(G)$ of at least ϕ, then $\widehat{n}^*$ satisfies $|n-\widehat{n}^*| \leq \frac{\varepsilon}{2-\varepsilon}n$ w.p. at least $1-2^{-\ell}$. The poof can be found in the full version article.*

Lower bound in terms of the general conductance

In the following we show that the bound of Observation 11 is almost tight. Recall that, given ϕ_ε, the modified version of Algorithm EdgeSampling in Observation 11 returns an estimate with and additive error of at most $\frac{\varepsilon}{2-\varepsilon}n$ using $O(n/\phi_\varepsilon)$ queries. In what follows we show that any algorithm, given the values ϕ_ε and ε, cannot output an estimate with an error smaller

[9] The definition of conductance in the directed setting is more involved and more importantly doesn't seem be directly relevant to the question of estimating the number of nodes. It suffers from similar problems as the skewed stationary distributions. Graphs having poor connectivity to a large fraction of the nodes may still have very *good* conductance if the total mass of the poorly connected nodes under the stationary distribution is very small.

Algorithm 1 EdgeSampling($\overrightarrow{\mathcal{O}}, \ell, \phi$).

$Y_0 = 0$ (fail surplus counter)
$v = (node)\ \overrightarrow{\mathcal{O}}.\mathsf{init}$ (query oracle to get the initial node)
$S_0 = \{v\}$
$E_0 = \{(v,i) \mid i \leq \deg^+(v)\}$ (set of undisclosed edges)
$t = 1$
while $Y_t \leq \ell$ **do**
  for $\tau = 1$ **to** $2/\phi$ **do**
    choose (u,i) uniformly at random from $E_{(t-1)\cdot 2/\phi+\tau-1}$.
    $v = (node)\ \overrightarrow{\mathcal{O}}.(u,i,\mathsf{out})$
    if $v \notin S_{t-1}$ **then**
      $S_t = S_{t-1} \cup \{v\}$
      $E_{(t-1)\cdot 2/\phi+\tau} \leftarrow (E_{(t-1)\cdot 2/\phi+\tau-1} \cup \{(v,i) \mid i \leq \deg^+(v)\}) \setminus \{(u,i)\}$
      break
    else
      $E_{t\cdot 2/\phi} \leftarrow E_{t\cdot 2/\phi-1} \setminus \{(u,i)\}$
  if $|S_t| = |S_{t-1}| + 1$ **then**
    $Y_t \leftarrow Y_t - 1$
  else
    $Y_t \leftarrow Y_t + 1$
    $S_t \leftarrow S_{t-1}$
  $t \leftarrow t + 1$
Output $|S_t|$.

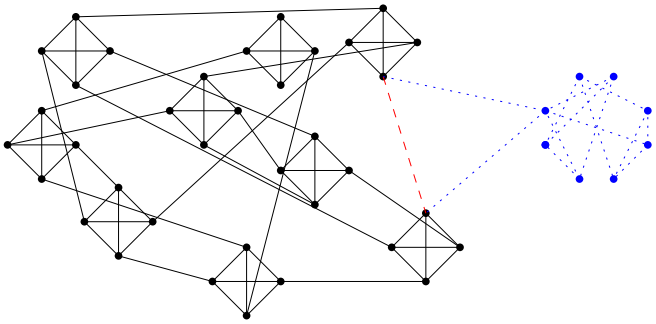

Figure 4 The graphs of Theorem 12. G contains the black nodes and the black and red (dashed) edges which form a $\lceil 1/\phi \rceil$-regular expander on cliques of size $\lceil 1/\phi \rceil$. G' is obtained by removing the red (dashed) edge and adding the blue (dotted) graph. At least one blue edge needs to be sampled, which takes $\Omega(n/\phi)$ time, in order to estimate n accurately.

than $\frac{\varepsilon-\delta}{2-\varepsilon-\delta}n$ unless it makes $\Omega(n/\phi_\varepsilon)$ queries, for any $\delta < \varepsilon/2$. We prove the following theorem for undirected graphs using $\mathcal{O}$, but it should be clear that the same result holds for directed graphs, by making the graph directed, with symmetric edges, and using $\overrightarrow{\mathcal{O}}(2)$ (and hence also for oracles $\overrightarrow{\mathcal{O}}$ and $\overrightarrow{\mathcal{O}}(1)$).

▶ **Theorem 12.** *Let $n \in \mathbb{N}$, $\phi \in [1/n, 1]$ and $\varepsilon \in (0, 1/2]$. There exists an undirected graph with general conductance $\phi_\varepsilon = \Theta(\phi)$ such that any algorithm with access to $\mathcal{O}$ requires $\Omega(n/\phi_\varepsilon)$ queries to output $\widehat{n}$ such that $|n - \widehat{n}| \leq \frac{\varepsilon-\delta}{2-\varepsilon-\delta}n$ w.p. at least $2/3$ for any $\delta < \varepsilon/2$.*

References

1 Michael A. Bender and Data Ron. Testing properties of directed graphs: Acyclicity and connectivity. *Random Structures and Algorithms*, 20(2):184–205, 2002.

2 Béla Bollobás. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European Journal of Combinatorics*, 1(4):311–316, 1980. doi:10.1016/S0195-6698(80)80030-8.

3 Mickey Brautbar and Michael Kearns. Local algorithms for finding interesting individuals in large networks. In *In Proceedings of ICS 2010*, pages 188–199, 2010.

4 Colin Cooper and Alan Frieze. Crawling on web graphs. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 419–427. ACM, 2002.

5 Colin Cooper, Tomasz Radzik, and Yiannis Siantos. Estimating network parameters using random walks. In *Computational Aspects of Social Networks, 4th International Conference on*, pages 33–40, 2012.

6 Artur Czumaj, Pan Peng, and Christian Sohler. Relating two property testing models for bounded degree directed graphs. In *Proceedings of the ACM Symposium on the Theory of Computing (STOC)*, 2016.

7 Anirban Dasgupta, Ravi Kumar, and Tamás Sarlós. On estimating the average degree. In *Proceedings of the 23rd international conference on World Wide Web*, pages 795–806. ACM, 2014. doi:10.1145/2566486.2568019.

8 Mark Finkelstein, Howard G Tucker, and Jerry Alan Veeh. Confidence intervals for the number of unseen types. *Statistics & Probability Letters*, 37(4):423–430, 1998.

9 Oded Goldreich. Introduction to testing graph properties. Survey article available at http://www.wisdom.weizmann.ac.il/~oded/COL/tgp-intro.pdf, 2010.

10 I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3/4):237–264, 1953.

11 Varun Kanade, Frederik Mallmann-Trenn, and Victor Verdugo. How large is your graph? *CoRR*, abs/1702.03959, 2017. URL: http://arxiv.org/abs/1702.03959.

12 Liran Katzir and Stephen J. Hardiman. Estimating clustering coefficients and size of social networks via random walk. *ACM Transactions on the Web*, 9(4):19:1–19:20, 2015.

13 Liran Katzir, Edo Liberty, Oren Somekh, and Ioana A. Cosma. Estimating sizes of social networks via biased sampling. *Internet Mathematics*, 10(3-4):335–359, 2014. doi:10.1080/15427951.2013.862883.

14 Donald E. Knuth. Estimating the efficiency of backtrack programs. *Mathematics of computation*, 29(129):122–136, 1975.

15 Alberto Marchetti-Spaccamela. On the estimate of the size of a directed graph. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 317–326. Springer, 1988.

16 Milena Mihail, Amin Saberi, and Prasad Tetali. Random walks with lookahead on power law random graphs. *Internet Mathematics*, 3(2):147–152, 2006.

17 Cameron Musco, Hsin-Hao Su, and Nancy A. Lynch. Ant-inspired density estimation via random walks: Extended abstract. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*, pages 469–478, 2016.

18 Leonard Pitt. A note on extending knuth's tree estimator to directed acyclic graphs. *Information processing letters*, 24(3):203–206, 1987.

19 Gregory Valiant and Paul Valiant. Estimating the unseen: an n/log(n)-sample estimator for entropy and support size, shown optimal via new clts. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 685–694. ACM, 2011.

20 Gregory Valiant and Paul Valiant. Estimating the unseen: improved estimators for entropy and other properties. In *Advances in Neural Information Processing Systems*, pages 2157–2165, 2013.