On Estimating the Average Degree of a Graph

Oded Goldreich*
Department of Computer Science
Weizmann Institute of Science
Rehovot, ISRAEL.
oded.goldreich@weizmann.ac.il

Dana Ron†
Department of EE–Systems
Tel-Aviv University
Ramat-Aviv, ISRAEL.
danar@eng.tau.ac.il

February 10, 2004

Abstract

Following Feige, we consider the problem of estimating the average degree of a graph. Using "neighbor queries" as well as "degree queries", we show that the average degree can be approximated arbitrarily well in sublinear time, unless the graph is extremely sparse (e.g., unless the graph has a sublinear number of edges).

We also provide an alternative proof of a result that is almost as good as Feige's; that is, we show that a 2-approximation of the average degree can be obtained in sublinear time in a model that only allows degree queries.

Keywords: Sublinear-time algorithms, randomized approximation algorithms

*Work done while being a fellow of the Radcliffe Institute for Advanced Study, Harvard University.
†Work done while being a fellow of the Radcliffe Institute for Advanced Study, Harvard University.

1 Introduction

In a recent work [2], Feige investigated the problem of estimating the average degree of a graph when given direct access to the list of degrees (of individual vertices). He observed two interesting ("phase transition") phenomena. Firstly, in contrast to the problem of estimating the average value of an arbitrary function $d : [N] \to \{1, ..., N-1\}$, sublinear-time approximations can be obtained when the function represents the degree sequence of a simple graph.[1] Secondly, whereas a $(2+\epsilon)$-approximation can be obtained in $O(\sqrt{|V|})$-time, for every constant $\epsilon > 0$, a better approximation factor cannot be achieved in sublinear time (i.e., a $(2 - o(1))$-approximation requires time $\Omega(|V|)$).

When viewing the problem of estimating the average degree in a graph as a special case of the problem of estimating the average value of an arbitrary function $d : [N] \to \{1, ..., N-1\}$, it seems natural to restrict the algorithm to "degree queries". However, from the point of view of sublinear-time algorithms for graphs (cf., e.g., [3, 4, 6, 1, 5]), it seems natural to allow also other queries to the graph (e.g., neighbor queries). Specifically, we augment the model by allowing a weak form of neighbor queries; that is, we consider a model in which one may obtain a *random* neighbor of any vertex of one's choice. (This model is weaker than the standard neighbor query model, investigated in [3, 5] and other works, in which the queries of the form (v, i) are allowed and are answered with the i^{th} neighbor of v.)

Our main result is a sublinear algorithm (in the augmented model) that obtains an arbitrarily good approximation of the average degree. Specifically, for every constant $\epsilon > 0$, we obtain a $(1+\epsilon)$-approximation in $\widetilde{O}(\sqrt{|V|})$-time. More precisely, the running time is $\widetilde{O}(\sqrt{|V|}) \cdot \text{poly}(1/\epsilon)$.

Turning back to the bare model (of Feige [2]), we provide an alternative proof of his result. That is, for every $\epsilon > 0$, we obtain a $(2+\epsilon)$-approximation (in the bare model) in $\widetilde{O}(\sqrt{|V|})$-time.[2] We believe that our analysis sheds more light on the reason that the approximation factor cannot be better than two. This alternative analysis, which is inspired by the work of [5], is also instructive as a warm-up toward our main result.

The above represents a simplified account of the results. We recall that Feige [2] has provides his algorithm with a lower-bound on the average degree of the input graph. This auxiliary input allows also to handle graphs that have isolated vertices (rather than assuming that each vertex has degree at least 1) and yields an improvement whenever the lowerbound is better (than the obvious value of 1). Specifically, given a lowerbound of ℓ (on the average degree), the complexity of Feige's algorithm is related to $\sqrt{|V|/\ell}$ rather than to $\sqrt{|V|}$. The same improvement holds also for our algorithms. Furthermore, we observe that our algorithms (as well as Feige's) can be adapted to work without this lowerbound. Specifically, the complexity of the modified algorithm, which obtains no a priori information about the average degree, is related to $\sqrt{|V|/\overline{d}}$ where $\overline{d}$ denotes the actual average degree (which is, of course, not given to the algorithm).

[1] Here we also assume that there are no isolated vertices in the graph (i.e., each vertex has degree at least 1).

[2] The running-time of Feige's algorithm is slightly better: His algorithm uses $O(\sqrt{|V|}/\epsilon)$ samples, whereas our algorithm uses $t(|V|/\epsilon) \cdot \sqrt{|V|}/\epsilon^{3.5}$ samples, where $t(M) = O(\log M \cdot \log\log M)$.

The said adaptation is based on the following observation: Even when given a wrong lowerbound (on the average degree), none of the said algorithms outputs a (gross) overestimation of the average degree (except with small probability). Thus, we may iteratively run the algorithm starting with $\ell = |V|/2$ and decreasing ℓ by a factor of 2 in each iteration, until we obtain an output that is larger than the current value of ℓ. The point is that the algorithm's output is always an approximately correct lower-bound on the average degree, and so if this value is smaller than ℓ then the latter is a valid lower-bound on the average degree.

2 The Algorithms

Let $G = (V, E)$ be a simple graph (i.e., having no parallel edges and no self-loops), and let $d(v)$ denote the degree of vertex $v \in V$ in G (i.e., $d(v) \stackrel{\text{def}}{=} |\{u : \{u, v\} \in E\}|$). We denote by $\overline{d} \stackrel{\text{def}}{=} \sum_{v\in V} d(v)/|V|$ the average degree in G, and so $\overline{d} \cdot |V| = 2|E|$.

Both our algorithms can sample uniformly in V as well as obtain the value of $d(v)$ for any $v \in V$ of their choice (that is, perform degree-queries). First, in Subsection 2.1, we give a $(2+\epsilon)$-approximation procedure for $\overline{d}$ that uses only degree queries. Next, in Subsection 2.2, we modify this procedure in order to obtain a $(1 + \epsilon)$-approximation procedure. This procedure uses "random neighbor" queries in addition to degree queries. Both procedures take as input a parameter, ℓ, which is an *a priori known lowerbound* on $\overline{d}$. The larger ℓ, the more efficient these algorithms are. Finally, in Subsection 2.3, we eliminate the need for this a priori knowledge.

2.1 The Basic Procedure

The following procedure only uses the ability to obtain the degree of uniformly selected vertices. That is, it merely requires the ability to sample vertices at random and the ability to obtain the degree of any desired vertex. The procedure refers to two parameters $\epsilon, \beta > 0$, where ϵ is the main approximation parameter and β may as well equal $\epsilon/4$.

Factor-2 Approximation Algorithm

1. Uniformly and independently select $K = \text{poly}((\log|V|)/\epsilon) \cdot \sqrt{|V|/\ell}$ vertices from V, and let S denote the (multi-)set of selected vertices.[a]
2. For $i = 0, 1, \ldots, \lceil \log_{(1+\beta)} |V| \rceil$, let $S_i = \left\{ v \in S : \ d(v) \in \left((1+\beta)^{i-1}, (1+\beta)^i \right] \right\}$.
3. Let $I = \left\{ i : \frac{|S_i|}{|S|} < \frac{1}{t} \cdot \sqrt{\frac{3\epsilon}{8} \cdot \frac{\ell}{|V|}} \right\}$, where $t \stackrel{\text{def}}{=} \lceil \log_{(1+\beta)} |V| \rceil + 1$.
4. Output $\frac{1}{K} \cdot \sum_{i \notin I} |S_i| \cdot (1+\beta)^i$.

[a] Setting $K = \widetilde{O}(\epsilon^{-3.5} \log|V|) \cdot \sqrt{|V|/\ell}$ will do.

In other words, this procedure outputs the average value of $(1+\beta)^{\lceil \log_{(1+\beta)} d(v) \rceil}$, taken over all $v \in S$ that belong to sufficiently large subsets S_i. Clearly, we might as well output the average value $d(v)$, taken over all $v \in S$ that belong to sufficiently large subsets S_i. Indeed, it would have been simpler to just output the average value of $d(v)$ over all $v \in S$, but our analysis refers to the former (whereas the analysis of Feige [2] refers to the latter). We comment that the above procedure and its analysis follow the ideas underlying the edge-sampling procedure of Kaufman *et. al.* [5].

Intuitively, assuming that $\frac{|S_i|}{|S|} \approx \frac{|B_i|}{|V|}$ (for all i's), it holds that

$$\frac{1}{K} \cdot \sum_i |S_i| \cdot (1+\beta)^i \;\approx\; \frac{1}{|V|} \cdot \sum_i |B_i| \cdot (1+\beta)^i \;\approx\; \overline{d} \tag{1}$$

The problem is that we may not obtain good approximations for all B_i's; in particular, we cannot obtain a good approximation for small B_i's (i.e., B_i's of size smaller than $|V|/K$). This is the reason that we discard the corresponding small S_i's. Consequently, our (implicit) double-counting of edges (which is used as an estimate for $\overline{d}|V|$) is biased as follows: Edges with both endpoints in small B_i's are not counted at all, edges with a single endpoint in a small B_i are counted once, and edges with both endpoints in big B_i's are counted twice (as they should). Thus, the above estimate of $\overline{d}|V|$ is never a gross overestimate. The underestimation is due to edges with at least one endpoint in a small B_i. For edges with a single endpoint in a small B_i we lose a factor of two, whereas the number of edges with both endpoints in small B_i's is upperbounded by the square of the number of vertices residing in small B_i's, which in turn is at most $\sqrt{(\epsilon/2) \cdot \ell |V|} \leq \sqrt{(\epsilon/2) \cdot \overline{d}|V|}$.

Theorem 1 *For every $\epsilon < 1/2$ and $\beta \leq \epsilon/4$, the above procedure outputs a value $\widetilde{\mathtt{d}}$ such that, with probability at least $2/3$, it holds that $(0.5-\epsilon) \cdot \overline{d} < \widetilde{\mathtt{d}} < (1+\epsilon) \cdot \overline{d}$.*

For arbitrary $\beta > 0$, we get $(0.5-\epsilon) \cdot \overline{d} < \widetilde{\mathtt{d}} < (1+(\epsilon/2)+2\beta) \cdot \overline{d}$.

Proof: Recall that $t = \lceil \log_{(1+\beta)} |V| \rceil + 1$, and define a partition of V into the following buckets:

$$B_i = \left\{ v : \; d(v) \in \left((1+\beta)^{i-1}, (1+\beta)^i \right] \right\}, \quad \text{for } i = 0, 1, \ldots, t-1 \; . \tag{2}$$

By definition of these buckets, it holds that

$$\overline{d} \;\leq\; \frac{1}{|V|} \sum_{i=1}^{t} |B_i| \cdot (1+\beta)^i \;\leq\; (1+\beta) \cdot \overline{d} \, . \tag{3}$$

Let $\rho \stackrel{\rm def}{=} (1/t)\sqrt{(\epsilon/4) \cdot \ell/|V|}$ be a density threshold. By our choice of the sample size K (which guarantees that $(\epsilon/2)^2 \cdot \rho \cdot K \geq 10 \log t$), with probability at least $1-(1/100t)$, the number of samples residing in any specific bucket having density above the threshold ρ is approximately proportional to the bucket's density, whereas few samples resides in buckets having density below the threshold. More precisely, with probability at least 0.99,

$$\forall i \;\text{ s.t. }\; |B_i| \geq \rho \cdot |V| : \qquad \left(1 - \frac{\epsilon}{2}\right) \cdot \frac{|B_i|}{|V|} \;\leq\; \frac{|S_i|}{K} \;\leq \left(1 + \frac{\epsilon}{2}\right) \cdot \frac{|B_i|}{|V|} \tag{4}$$

and

$$\forall i \text{ s.t. } |B_i| < \rho \cdot |V| : \qquad \frac{|S_i|}{K} < \frac{1}{t}\sqrt{(\epsilon/2) \cdot \ell/|V|} \tag{5}$$

In particular, in the latter case we have that $i \in I$. Eq. (3), (4) & (5) imply that, with high constant probability, it holds that

$$\begin{aligned} \widetilde{\mathtt{d}} \;=\; \frac{1}{K} \cdot \sum_{i \notin I} |S_i| \cdot (1+\beta)^i \;&\leq\; \frac{1}{|V|} \sum_{i=1}^{t} \left(1 + \frac{\epsilon}{2}\right) \cdot |B_i| \cdot (1+\beta)^i && (6) \\ &\leq\; \left(1 + \frac{\epsilon}{2}\right) \cdot (1+\beta) \cdot \overline{d}\,. && (7) \end{aligned}$$

Hence (for, say, $\beta = \epsilon/4$) we don't get too much of an overestimate; that is, $\widetilde{\mathtt{d}} \leq (1+\epsilon) \cdot \overline{d}$.

Turning to the possibility of underestimation, we first note that with high probability

$$\widetilde{\mathtt{d}} \;=\; \frac{1}{K} \cdot \sum_{i \notin I} |S_i| \cdot (1+\beta)^i \;\geq\; \frac{1}{|V|} \sum_{i \notin I} \left(1 - \frac{\epsilon}{2}\right) \cdot |B_i| \cdot (1+\beta)^i \tag{8}$$

It remains to lower bound the latter expression. We use the following notations and facts:

- For each i, let $E_i \stackrel{\text{def}}{=} \{(u,v) : u \in B_i \;\&\; \{u,v\} \in E\}$; that is, E_i is the set of *ordered* pairs of adjacent vertices such that the first vertex is in B_i. Thus, the E_i's are disjoint and *each edge contributes two pairs to the set* $\bigcup_i E_i$. Also,

$$|B_i| \cdot (1+\beta)^{i-1} \;<\; |E_i| \;\leq\; |B_i| \cdot (1+\beta)^i \tag{9}$$

- Let $U \stackrel{\text{def}}{=} \{v \in B_i : \; i \in I\}$ denote the set of vertices that reside in buckets that are deemed small by the sample S. Thus, with high probability,

$$|U| \;\leq\; \left|\left\{v \in B_i : \; |B_i| \leq (1/t) \cdot \sqrt{(\epsilon/2) \cdot |V| \cdot \ell}\right\}\right| \;\leq\; \sqrt{(\epsilon/2) \cdot |V| \cdot \ell} \tag{10}$$

  and $|U|^2 \leq (\epsilon/2) \cdot |V| \cdot \ell$.

- Let $E(V_1, V_2)$ denote the set of edges with one endpoint in V_1 and one endpoint in V_2. We consider a partition of $E(V,V)$ into the sets $E(V \setminus U, V \setminus U)$, $E(V \setminus U, U)$ and $E(U,U)$. Thus:

$$\begin{aligned} \overline{d}|V| \;&=\; 2|E(V \setminus U, V \setminus U)| + 2|E(V \setminus U, U)| + 2|E(U,U)| && (11) \\ &\leq\; 2|E(V \setminus U, V \setminus U)| + 2|E(V \setminus U, U)| + |U|^2 && (12) \end{aligned}$$

The key observation is that *the number of times that an edge is counted in the sum* $\sum_{i \notin I} |E_i|$ *equals the number of endpoints of the edge that reside in* $V \setminus U = \bigcup_{i \notin I} B_i$. Specifically, an edge with both endpoints in $V \setminus U$ is counted twice, an edge with a single endpoint in $V \setminus U$ is counted once, and an edge with both endpoints in U is not counted at all. Thus,

$$\sum_{i \notin I} |E_i| \;=\; 2|E(V \setminus U, V \setminus U)| + |E(V \setminus U, U)| \tag{13}$$

Combining Eq. (8), Eq. (9) and Eq. (13), it follows that

$$\widetilde{\mathtt{d}} \geq \frac{1}{|V|}\sum_{i\notin I}\left(1-\frac{\epsilon}{2}\right)\cdot|B_i|\cdot(1+\beta)^i \tag{14}$$

$$\geq \frac{1}{|V|}\sum_{i\notin I}\left(1-\frac{\epsilon}{2}\right)\cdot|E_i| \tag{15}$$

$$= \frac{1-(\epsilon/2)}{|V|}\cdot\Big(2|E(V\setminus U,V\setminus U)|+|E(V\setminus U,U)|\Big)\,. \tag{16}$$

Using Eq. (12) and Eq. (10), we get

$$\widetilde{\mathtt{d}} \geq \frac{1-(\epsilon/2)}{|V|}\cdot\Big(|E(V\setminus U,V\setminus U)|+|E(V\setminus U,U)|\Big) \tag{17}$$

$$\geq \frac{1-(\epsilon/2)}{2}\cdot\left(\overline{d}-\frac{|U|^2}{|V|}\right) \tag{18}$$

$$\geq \frac{1-(\epsilon/2)}{2}\cdot\overline{d}-\frac{\epsilon}{4}\cdot\ell \tag{19}$$

Since $\ell \leq \overline{d}$, we are done. ■

2.2 The Improved Procedure

Here we present an improved approximation in a model that allows, in addition to the "degree queries" used above, also "neighbor queries" (as in [3, 5] and other works). Namely, for any given vertex v and for any $j \leq d(v)$, we can obtain the j^{th} neighbor of v. Actually, it suffices to be able to obtain a random neighbor of any desired/queried vertex.

The improved procedure builds on the basic procedure (factor-2 approximation) presented in Section 2.1. Below we refer to the notations introduced in the proof of Theorem 1. The idea is to modify the basic procedure by estimating the fraction of edges in $E(V\setminus U,U)$, and compensating for the fact that these edges are counted only once in $\sum_{i\notin I}|E_i|$ by counting these edges twice. To this end, we take a slightly larger sample; specifically, the sample size K is chosen to be sufficiently large so that for each $i \notin I$ it holds that $|S_i| \geq L = \text{poly}((\log|V|)/\epsilon)$. For each $i \notin I$ and for each $v \in S_i$ we take a random neighbor of v and check whether it falls in $\bigcup_{j\in I} B_j$ (i.e., whether it belongs to U). We let $\widetilde{\alpha}_i$ be the fraction of neighbors (among the selected $|S_i|$ neighbors) that belong to U. Finally, in Step 4 of the algorithm, we replace $|S_i|\cdot(1+\beta)^i$ with $(1+\widetilde{\alpha}_i)|S_i|\cdot(1+\beta)^i$. The modified algorithm proceeds as follows:

Factor-$(1+\epsilon)$ Approximation Algorithm

Steps 1–3: As in the basic procedure, with $K = \widetilde{O}(L/\rho\epsilon^2)$ (rather than $K = \widetilde{O}((\log|V|)/\rho\epsilon^2)$), where (as before) $\rho \stackrel{\rm def}{=} (1/t)\sqrt{(\epsilon/4)\cdot \ell/|V|}$.

Additional Step: For every $i \notin I$ and every $v \in S_i$, select at random a neighbor u of v, and let $\chi(v) = 1$ if $u \in \bigcup_{j\in I} B_j$, and $\chi(v) = 0$ otherwise. For every $i \notin I$, let $\widetilde{\alpha}_i = |\{v \in S_i : \chi(v)=1\}|/|S_i|$.

Step 4 (modified): Output $\frac{1}{K}\cdot\sum_{i\notin I}(1+\widetilde{\alpha}_i)\cdot|S_i|\cdot(1+\beta)^i$.

Theorem 2 *For every $\epsilon < 1/2$ and $\beta \le \epsilon/4$, the modified procedure outputs a value $\widetilde{\mathtt{d}}$ such that, with probability at least $2/3$, it holds that $(1-\epsilon)\cdot\overline{d} < \widetilde{\mathtt{d}} < (1+\epsilon)\cdot\overline{d}$.*

Proof: Let $E'_i = \{(u,v) \in E_i : \ v \in U\}$ and $\alpha_i = |E'_i|/|E_i|$. That is, for $i \notin I$, the set E'_i contains pairs of adjacent vertices with a single endpoint in $V \setminus U$, and the corresponding edge is counted only once in the sum $\sum_{i\notin I}|E_i|$ (similarly for $\sum_{i\notin I}|E'_i|$). Thus:

$$\sum_{i\notin I}|E'_i| \;=\; |E(V\setminus U, U)| \tag{20}$$

$$\sum_{i\notin I}|E_i\setminus E'_i| \;=\; 2|E(V\setminus U, V\setminus U)| \tag{21}$$

By our choice of L we have that with high probability, for every $i \notin I$ such that $\alpha_i \ge \epsilon/8$,

$$\left(1-\frac{\epsilon}{4}\right)\cdot\alpha_i \;\le\; \widetilde{\alpha}_i \;\le\; \left(1+\frac{\epsilon}{4}\right)\cdot\alpha_i \tag{22}$$

and if $\alpha_i < \epsilon/8$ then $\widetilde{\alpha}_i < \epsilon/4$.

Thus, with high probability, all approximations are quite good, in which case the following holds:

$$\begin{aligned}
\widetilde{\mathtt{d}} &= \frac{1}{K}\cdot\sum_{i\notin I}(1+\widetilde{\alpha}_i)\cdot|S_i|\cdot(1+\beta)^i && (23)\\
&\le \frac{1}{|V|}\cdot\sum_{i\notin I}(1+\widetilde{\alpha}_i)\cdot\left(1+\frac{\epsilon}{2}\right)\cdot|B_i|\cdot(1+\beta)^i && (24)\\
&\le \frac{1+\epsilon/2}{|V|}\cdot\Bigg(\sum_{\substack{i\notin I\\ \alpha_i\ge\epsilon/8}}(1+(1+\epsilon/4)\alpha_i)\cdot(1+\beta)\cdot|E_i| \\
&\qquad\qquad + \sum_{\substack{i\notin I\\ \alpha_i<\epsilon/8}}(1+\epsilon/4)\cdot(1+\beta)\cdot|E_i|\Bigg) && (25)\\
&\le \frac{(1+\epsilon)\cdot(1+\beta)}{|V|}\cdot\sum_{i\notin I}(1+\alpha_i)\cdot|E_i| && (26)
\end{aligned}$$

where Eq. (25) uses Eq. (9) and our assumption on the estimates $\widetilde{\alpha}_i$. Similarly,

$$\widetilde{\mathtt{d}} \geq \frac{(1-\epsilon)}{|V|} \cdot \sum_{i \notin I} (1+\alpha_i) \cdot |E_i| \tag{27}$$

Using $\beta \leq \epsilon/4$ and $|E'_i| = \alpha_i \cdot |E_i|$, we have

$$\begin{aligned}
\widetilde{\mathtt{d}} &= \frac{1 \pm (3\epsilon/2)}{|V|} \cdot \sum_{i \notin I} (1+\alpha_i) \cdot |E_i| && (28)\\
&= \frac{1 \pm (3\epsilon/2)}{|V|} \cdot \Big(\sum_{i \notin I} |E_i \setminus E'_i| + \sum_{i \notin I} |E'_i| + \sum_{i \notin I} \alpha_i \cdot |E_i| \Big) && (29)\\
&= \frac{1 \pm (3\epsilon/2)}{|V|} \cdot \Big(\sum_{i \notin I} |E_i \setminus E'_i| + 2 \sum_{i \notin I} |E'_i| \Big) && (30)
\end{aligned}$$

Using Eq. (20) & (21), it follows that

$$\begin{aligned}
\widetilde{\mathtt{d}} &= \frac{1 \pm (3\epsilon/2)}{|V|} \cdot \Big(2|E(V \setminus U, V \setminus U)| + 2|E(V \setminus U, U)| \Big) && (31)\\
&= \frac{1 \pm (3\epsilon/2)}{|V|} \cdot \Big(2|E(V,V)| - 2|E(U,U)| \Big) && (32)\\
&= \frac{1 \pm (3\epsilon/2)}{|V|} \cdot \Big(\overline{d}|V| \pm |U|^2 \Big) && (33)
\end{aligned}$$

Recalling that $|U|^2 \leq (\epsilon/2) \cdot \ell|V|$ (and $\ell \leq \overline{d}$), we get $\widetilde{\mathtt{d}} = (1 \pm 2\epsilon) \cdot \overline{d}$. Substituting ϵ by $\epsilon/2$, the theorem follows. ■

2.3 Working without a degree lower-bound

For sake of simplicity, we start by modifying both our algorithms such that, when given a valid lower-bound ℓ, they do not output an overestimation of the average degree (except with small probability). This is done by simply decreasing the output by a factor of $1+\epsilon$. Thus, the output, $\widetilde{\mathtt{d}}$, of the modified first (resp., second) algorithm satisfies $\Pr[(0.5-2\epsilon)\overline{d} < \widetilde{\mathtt{d}} < \overline{d}] \geq 2/3$ (resp., $\Pr[(1-2\epsilon)\overline{d} < \widetilde{\mathtt{d}} < \overline{d}] \geq 2/3$). Furthermore, by $O(1) + \log\log|V|$ repetitions, we may reduce the probability of error to below $1/(6\log|V|)$.

An interesting feature of our algorithms is that, with high probability, they do not output an overestimate of $\overline{d}$ even in case they are invoked with a parameter ℓ that is *higher* than the average degree $\overline{d}$ (i.e., is not a valid lower-bound). To verify this feature, observe that the only place in the analysis where we rely on the assumption $\ell \leq \overline{d}$ is in bounding the underestimation error (i.e., when bounding the total number of edges with both endpoints in U). (We comment that also Feige's algorithm [2] has this feature, but for different reasons.)

This feature allows us to present versions of these algorithms that do not require an a priori lower-bound on the average degree. Specifically, let us denote by A_i the algorithm presented in Section 2.i, (where $i \in \{1,2\}$ and when the algorithm is modified as suggested above). Then, starting with $\ell = |V|/2$, we may proceed in at most $2\log_2|V|$ iterations as follows. We invoke A_i with the current value of ℓ, and let $\widetilde{\mathtt{d}}$ denote the output obtained. If $\widetilde{\mathtt{d}} \geq \ell$ then we halt and output $\widetilde{\mathtt{d}}$, otherwise we proceed to the next iteration while setting $\ell \leftarrow \ell/2$. In case all iterations were completed and still $\widetilde{\mathtt{d}} < \ell$ in the last iteration (i.e., $\widetilde{\mathtt{d}} < 1/2|V|$) then the graph must have no edges and we halt outputting $\widetilde{\mathtt{d}} = 0$.

Let $\ell_j = |V|/2^j$ be the parameter used in the j-th invocation of algorithm A_1 (resp., A_2), and let $\widetilde{\mathtt{d}}_j$ denote the corresponding output. Then, with probability at least $2/3$, for every iteration j that took place, it holds that $\widetilde{\mathtt{d}}_j \leq \overline{d}$ and if $\overline{d} \geq \ell_j$ then $\widetilde{\mathtt{d}}_j \geq (0.5-2\epsilon)\overline{d}$ (resp., $\widetilde{\mathtt{d}}_j \geq (1-2\epsilon)\overline{d}$). In this case, assuming the graph contains any edges at all,[3] the algorithm will stop after at most $\log(|V|/\overline{d}) + O(1)$ iterations, and will output a value that is in the interval $[(0.5-2\epsilon)\overline{d}, \overline{d}]$ (resp., the interval $[(1-2\epsilon)\overline{d}, \overline{d}]$). Thus, the overall running-time of the algorithm is $\mathrm{poly}(\epsilon^{-1}\log|V|) \cdot \sqrt{|V|/\overline{d}}$.

3 Concluding Remarks

We first observe that any constant approximation algorithm must perform $\Omega(\sqrt{N})$ queries (even when both degree queries and neighbor queries are allowed). To verify this, consider the following two graphs. One graph is simply a cycle over all vertices, so that the average degree is 2. The other graphs consists of a cycle of size $|V| - c \cdot \sqrt{|V|}$, for some constant c, and a clique of size $c \cdot \sqrt{|V|}$. This graph has average degree $2 + c - o(|V|)$. But for a sufficiently small constant c', if an algorithm performs less than $c' \cdot \sqrt{|V|}$ queries then it cannot distinguish between the two graphs. To be more precise, we need to consider two corresponding distributions on graphs (allowing all possible labelings of the graph vertices), so that $\Omega(\sqrt{|V|})$ queries are required to distinguish between a random graph from one distribution and a random graph from the other distribution.

We also note that if parallel edges (or weighted edges) are allowed, then estimating the average degree of a graph requires $\Omega(|V|)$ queries (even when both degree queries and neighbor queries are allowed). Consider the following two graphs (families of graphs): one graph consists of a cycle over all vertices (with a single edge between every pair of consecutive vertices) and the other consists of a cycle over $|V|-2$ vertices, and a pair of vertices with $c \cdot |V|$ parallel edges between them. The average degree in the first graph is 2 whereas the average degree in the second graph is roughly $2 + c$. But distinguishing between the two (families of) graphs requires $\Omega(|V|)$ queries. Thus there is a gap between the query complexity of estimating the average degree of simple graphs and non-simple graphs.

[3] In case the graph contains no edges, the algorithm will complete all iterations with no output (because $\overline{d} = 0 < \ell_j$ whereas $\widetilde{\mathtt{d}}_j = 0$ for each $j \leq 2\log|V|$), and thus output the correct value (i.e., 0) at the last step. In this case, the overall running-time of the algorithm is $\mathrm{poly}(\epsilon^{-1}\log|V|) \cdot |V|$. Clearly one can modify the algorithm so that its complexity is never more that $O(|V|)$ (i.e., the complexity of computing the exact average degree), by stopping once ℓ_j goes below $\mathrm{poly}(\epsilon^{-1}\log|V|)/|V|$ for an appropriate polynomial in $\log|V|$ and ϵ^{-1}.

Acknowledgments

We are grateful to Uri Feige for pointing out an error in an earlier version of this work.

References

[1] B. Chazelle, R. Rubinfeld, and L. Trevisan. Approximating the Minimum Spanning Tree Weight in Sublinear Time. In *Proc. of the 28th ICALP*, pages 190–200, 2001.

[2] U. Feige. On sums of independent random variables with unbounded variance, and estimating the average degree in a graph. To appear in *Proc. of the 36th STOC*, 2004.

[3] O. Goldreich and D. Ron. Property Testing in Bounded Degree Graphs. *Algorithmica*, Vol. 32 (2), pages 302–343, 2002.

[4] O. Goldreich and D. Ron. A Sublinear Bipartitness Tester for Bounded Degree Graphs. *Combinatorica*, Vol. 19 (3), pages 335–373, 1999.

[5] T. Kaufman, M. Krivelevich, and D. Ron. Tight Bounds for Testing Bipartiteness in General Graphs. In *Proc. of RANDOM'03*, pages 341–353, 2003.

[6] M. Parnas and D. Ron. Testing the diameter of graphs. *Random Structures and Algorithms*, Vol. 20 (2), pages 165–183, 2002.