Detecting High Log-Densities – an $O(n^{1/4})$ Approximation for Densest k-Subgraph

Aditya Bhaskara * Moses Charikar † Eden Chlamtac ‡ Uriel Feige §
Aravindan Vijayaraghavan ¶

Abstract

In the Densest k-Subgraph problem, given a graph G and a parameter k, one needs to find a subgraph of G induced on k vertices that contains the largest number of edges. There is a significant gap between the best known upper and lower bounds for this problem. It is NP-hard, and does not have a PTAS unless NP has subexponential time algorithms. On the other hand, the current best known algorithm of Feige, Kortsarz and Peleg [FKP01], gives an approximation ratio of $n^{1/3-\varepsilon}$ for some specific $\varepsilon > 0$ (estimated by those authors at around $\varepsilon = 1/60$).

We present an algorithm that for every $\varepsilon > 0$ approximates the Densest k-Subgraph problem within a ratio of $n^{1/4+\varepsilon}$ in time $n^{O(1/\varepsilon)}$. If allowed to run for time $n^{O(\log n)}$, our algorithm achieves an approximation ratio of $O(n^{1/4})$. Our algorithm is inspired by studying an average-case version of the problem where the goal is to distinguish random graphs from random graphs with planted dense subgraphs – the approximation ratio we achieve for the general case matches the "distinguishing ratio" we obtain for this planted problem. Achieving a distinguishing ratio of $o(n^{1/4})$ for the planted problem (in polynomial time) is beyond the reach of our current techniques.

At a high level, our algorithms involve cleverly counting appropriately defined trees of constant size in G, and using these counts to identify the vertices of the dense subgraph. Our algorithm is based on the following principle. We say that a graph $G(V, E)$ has log-density α if its average degree is $\Theta(|V|^{\alpha})$. The algorithmic core of our result is a family of algorithms that output k-subgraphs of nontrivial density whenever the log-density of the densest k-subgraph is larger than the log-density of the host graph.

Finally, we extend this algorithm to obtain an $O(n^{1/4-\varepsilon})$-approximation algorithm which runs in time $O(2^{n^{O(\varepsilon)}})$ and also explore various approaches to obtain better approximation algorithms in restricted parameter settings for random instances.

*Department of Computer Science, Princeton University, supported by NSF awards MSPA-MCS 0528414, CCF 0832797, and AF 0916218. Email: bhaskara@cs.princeton.edu

†Department of Computer Science, Princeton University, supported by NSF awards MSPA-MCS 0528414, CCF 0832797, and AF 0916218. Email: moses@cs.princeton.edu

‡Weizmann Institute of Science, Rehovot, Israel, supported by a Sir Charles Clore Postdoctoral Fellowship. Email: eden.chlamtac@weizmann.ac.il

§Weizmann Institute of Science, Rehovot, Israel. Email: uriel.feige@weizmann.ac.il. The author holds the Lawrence G. Horowitz Professorial Chair at the Weizmann Institute. Work supported in part by The Israel Science Foundation (grant No. 873/08).

¶Department of Computer Science, Princeton University, supported by NSF awards MSPA-MCS 0528414, CCF 0832797, and AF 0916218. Email: aravindv@cs.princeton.edu

1 Introduction

In this paper, we study the Densest k-Subgraph (DkS) problem: Given a graph G and parameter k, find a subgraph of G on k vertices with maximum density (average degree). This problem may be seen as an optimization version of the classical NP-complete decision problem CLIQUE. The approximability of DkS is an important open problem and despite much work, there remains a significant gap between the currently best known upper and lower bounds.

In addition to being NP-hard (as seen by the connection to CLIQUE), the DkS problem has also been shown not to admit a PTAS under various complexity theoretic assumptions. Feige [Fei02] has shown this assuming random 3-SAT formulas are hard to refute, while more recently this was shown by Khot [Kho04] assuming that NP does not have randomized algorithms that run in sub-exponential time (i.e. that $NP \not\subseteq \cap_{\varepsilon>0} BPTIME(2^{n^\varepsilon})$).

The current best approximation ratio of $n^{1/3-\varepsilon}$ for some small $\varepsilon > 0$ (which has been estimated to be roughly 1/60) was achieved by Feige, Kortsarz and Peleg [FKP01]. Other known approximation algorithms have approximation guarantees that depend on the parameter k. The greedy heuristic of Asahiro et al. [AHI02] obtains an $O(n/k)$ approximation. Linear and semidefinite programming (SDP) relaxations were studied by Srivastav and Wolf [SW98] and by Feige and Langberg [FL01], where the latter authors show how they can be used to get approximation ratios somewhat better than n/k. Feige and Seltser [FS97] show graphs for which the integrality gap of the natural SDP relaxation is $\Omega(n^{1/3})$, indicating that in the worst case, the approximation ratios achieved in [FKP01] are better than those achievable by this SDP. When the input is a complete graph with edge weights that satisfy the triangle inequality, a simple greedy algorithm achieves an approximation ratio of 2 (though the analysis of this algorithm is apparently not easy, see [BG09]).

A related problem to DkS is the max density subgraph problem, where the aim is to find a subgraph H which maximizes the ratio of number of edges to number of vertices in H. It turns out that this can be solved in polynomial time [GGT89]. Charikar et al. [CHK09] recently showed an $O(n^{1/3})$ approximation to the maximization version of label cover. This problem is at least as difficult as DkS in the sense that there is an approximation preserving randomized reduction from DkS (see [CHK09] for example) to it. No reduction in the opposite direction is known.

Our algorithm for DkS is inspired by studying an average-case version we call the 'Dense vs Random' question (see Section 3 for a precise definition). Here the aim is to distinguish random graphs from graphs containing a dense subgraphs, which can be viewed as the task of efficiently certifying that random graphs do not contain dense subgraphs. This distinguishing problem is similar in flavour to the well-studied planted clique problem (see [AKS98]). Getting a better understanding of this planted question seems crucial for further progress on DkS.

Some recent papers have used the hypothesis that (bipartite versions of) the planted dense subgraph problem is computationally hard: Applebaum et al. [ABW08] use this in the design of a new public key encryption scheme. More recently, Arora et al. [ABBG10] use this to demonstrate that evaluating certain financial derivatives is computationally hard. The use of such hardness assumptions provides additional motivation for the study of algorithms for these problems.

1.1 Our results

Our main result is a polynomial time $O(n^{1/4+\varepsilon})$ approximation algorithm for DkS, for any constant $\varepsilon > 0$. That is, given $\varepsilon > 0$, and a graph G with a k-subgraph of density d, our algorithm outputs a k-subgraph of density $\Omega\left(d/n^{1/4+\varepsilon}\right)$ in polynomial time. In particular, our techniques give an

$O(n^{1/4})$-approximation algorithm running in $O(n^{\log n})$ time.

At a high level, our algorithms involve cleverly counting appropriately defined subgraphs of constant size in G, and use these counts to identify the vertices of the dense subgraph. A key notion which comes up in the analysis is the following:

Definition 1.1. The *log-density* of a graph $G(V, E)$ with average degree D is $\log_{|V|} D$. In other words, if a graph has log-density α, its average degree is $|V|^{\alpha}$. [1]

We first consider the random setting – distinguishing between G drawn from $G(n, p)$, and G containing a k-subgraph H of certain density planted in it. In fact, we examine a few variants (i.e. in the second case each of G and H may or may not be random). For all these variants we show that if the log-density of G is α and that of H is β, with $\beta > \alpha$, we can solve the distinguishing problem in time $n^{O(1/(\beta-\alpha))}$.

Our main technical contribution is that a result of this nature can be proven for arbitrary graphs. Informally, our main result, which gives a *family* of algorithms, parametrized by a rational number r/s, can be stated as follows (see Theorem 4.5 for a more precise statement):

Theorem 1.2. (informal) *Let $s > r > 0$ be relatively prime integers, let G be an undirected graph with maximum degree $D = n^{r/s}$, which contains a k-subgraph H with average degree d. Then there is an algorithm running in time $n^{O(r)}$ that finds a k-subgraph of average degree $\Omega(d/D^{(s-r)/s})$.*

Note that the log-density of H does not explicitly occur in the statement of the theorem. However, it turns out we can pre-process the graph, and restrict ourselves to the case $kD = n$ (see Appendix A.2), in which case $D^{(s-r)/s} = k^{r/s}$, thus the output subgraph has average degree $d/k^{r/s}$. So if the log-density of H is β (recall that G has log-density $\leq r/s$), the output graph has density $d/k^{r/s} = k^{\beta - r/s}$. Thus the difference in the log-densities also plays a role in the case of arbitrary graphs.

Also note that the theorem deals with the *maximum* degree in G, and not average degree (which defines the log-density). It turns out that this upper-bound on the log-density will suffice (and will be more useful in the analysis).

As we observed earlier, we give a family of algorithms parameterized by a rational number. Thus, given G and k, we pick r/s appropriately and appeal to the theorem. In some sense, this family of algorithms is a systematic generalization of the (somewhat ad hoc) algorithms of [FKP01].

Finally, observe that theorem implies an approximation ratio of at most $D^{(s-r)/s} \leq n^{r(s-r)/s^2} \leq n^{1/4}$ for every choice of $s > r > 0$. As we mentioned, the statement above is informal. If we choose to restrict the running time to $O(n^{s_0})$ by limiting ourselves to $r < s \leq s_0$ (i.e. the bound on D will not be exact), we lose a factor n^{1/s_0} in the approximation. We refer to Section 4 for the details.

Outline of techniques. The distinguishing algorithm for the Dense vs Random problem is based on the fact that in $G(n, p)$, instances of any fixed constant size structure appear more often when the graph has a higher log-density. More precisely, given parameters r, s, we will define a (constant size) tree $T_{r,s}$ such that a fixed set of leaves can be completed to many instances of $T_{r,s}$ in a graph with log-density $> r/s$, whereas in a random graph with log-density $< r/s$ there will only be a negligible number of such instances. Thus, if the log-density of H is greater than r/s, finding a small set of vertices in H (and using them as leaves of $T_{r,s}$) can help reveal larger portions of a dense

[1] We will ignore low order terms when expressing the log-density. For example, graphs with constant average degree will be said to have log-density 0, and cliques will be said to have log-density 1.

subgraph. Though our intuition comes from random graphs, the heart of the argument carries over to worst-case instances.

We use a linear programming relaxation to guide us in our search for the fixed vertices' assignment and obtain the dense subgraph. In order to extract consistent feasible solutions from the LP even under the assumptions that fixed vertices belong to H, the LP relaxation will have a recursive structure similar to the Lovász-Schrijver hierarchy [LS91]. Feasible solutions to this LP (when they exist) can be found in time $n^{O(r)}$ (where the depth of the recursion will be roughly r), while the rest of the algorithm (given the LP solution) will take linear time. As we shall see, there is also a combinatorial variant of our algorithm, which, rather than relying on an LP solution, finds the appropriate set of leaves by exhaustive search (in time $n^{r+O(1)}$). While the analysis is essentially the same as for the LP variant, it is our hope that a mathematical programming approach will lead to further improvements in running time and approximation guarantee.

The approximation ratio we achieve for general instances of DkS matches the "distinguishing ratio" we are currently able to achieve for various random settings. This suggests the following concrete open problem which seems to be a barrier for obtaining an approximation ratio of $n^{1/4-\varepsilon}$ for DkS – distinguish between the following two distributions:

> $\mathcal{D}_1$: graph G picked from $G(n, n^{-1/2})$, and
> $\mathcal{D}_2$: graph G picked from $G(n, n^{-1/2})$ with the induced subgraph on $\sqrt{n}$ vertices replaced with $G(\sqrt{n}, n^{-(1/4+\varepsilon)})$.

In section 5 we will see that this distinguishing problem can be solved in time $2^{n^{O(\varepsilon)}}$, and this can be used to give an algorithm for DkS with approximation ratio $n^{1/4-\varepsilon}$, and run time $2^{n^{O(\varepsilon)}}$. These mildly exponential algorithms are interesting given the recent results of [ABBG10] and [ABW08], which are based on the assumption that planted versions of DkS are hard.

In section 6, we show that in the random setting we can beat the log-density based algorithms for certain ranges of parameters. We use different techniques for different random models, some of which are very different from those used in sections 3 and 4. Interestingly, none of these techniques give a distinguishing ratio better than $n^{1/4}$ when $k = D = \sqrt{n}$.

1.2 Organization of paper

In Section 2, we introduce some notation, and describe simplifying assumptions which will be made in later sections (some of these were used in [FKP01]). In Section 3, we consider two natural 'planted' versions of DkS, and present algorithms for these versions. The analysis there motivates our approximation algorithm for DkS, which will be presented in Section 4. In Section 5 and Section 6, we explore approaches to overcome the log-density barrier that limits our algorithms in Section 4. In Section 5 we give an $O(n^{1/4-\varepsilon})$ approximation algorithm for arbitrary graphs with run time $O(2^{n^{O(\varepsilon)}})$ time, and in Section 6, we show that in various random settings, we can obtain a $\sqrt{D}$-approximation (which is better than the log-density guarantee for $1 < D < \sqrt{n}$).

2 Notation and some simplifications

We now introduce some notation which will be used in the rest of the paper. Unless otherwise stated, $G(V, E)$ refers to an input graph on n vertices, and k refers to the size of the subgraph we are required to output. Also, H will denote the densest k-subgraph (breaking ties arbitrarily) in

G, and d denotes the average degree of H. For $v \in V$, $\Gamma(v)$ denotes the set of neighbors of v, and for a set of vertices $S \subseteq V$, $\Gamma(S)$ denotes the set of all neighbors of vertices in S. Finally, for any number $x \in \mathbb{R}$, will use the notation $\mathrm{fr}(x) = x - \lfloor x \rfloor$.

We will make the following simplifying assumptions in the remaining sections: (these are justified in Section A of the appendix)

1. There exists a D such that (a) the maximum degree of G is at most D, and (b) a greedy algorithm finds a k-subgraph of density $\max\{1, kD/n\}$ in G.
2. d is the minimum degree in H (rather than the average degree)
3. It suffices to find a subgraph of size *at most* k, rather than exactly k. In Section 4 we use 'k-subgraph' more loosely to mean a subgraph on at most k vertices.
4. When convenient, we may also take G (and hence H) to be bipartite.
5. The edges of the graph G are assumed to be unweighted, since we can bucket the edges into $O(\log n)$ levels according to the edge weights (which we assume are all positive), and output the densest of the k-subgraphs obtained by applying the algorithm to each of the graphs induced by the edges in a bucket. This incurs a loss of just $O(\log n)$ factor in the approximation.

In many places, we will ignore leading constant factors (for example, we may find a subgraph of size $2k$ instead of k). It will be clear that these do not seriously affect the approximation factor.

3 Random graph models

An $f(n)$-approximation algorithm for the densest k-subgraph problem must be able to distinguish between graphs where any k-subgraph has density at most c, and graphs with an $cf(n)$-dense k-subgraph planted in them. Random graphs are a natural class of graphs that do not contain dense k-subgraphs. Further, random graphs seem to present challenges for currently best known algorithms for DkS. Hence, it is instructive to see what parameters allow us to efficiently solve this distinguishing problem.

We consider three variants of the random distinguishing problem, in increasing order of difficulty. In the *Random Planted Model*, we would like to distinguish between two distributions:

$\mathcal{D}_1$: Graph G is picked from $G(n,p)$, with $p = n^{\alpha-1}$, $0 < \alpha < 1$.
$\mathcal{D}_2$: G is picked from $G(n, n^{\alpha-1})$ as before. A set S of k vertices is chosen arbitrarily, and the subgraph on S is replaced with a random graph H from $G(k, k^{\beta-1})$ on S.

A slightly harder variant is the *Dense in Random* problem, in which we would like to distinguish between G chosen from $\mathcal{D}_1$, as before, and G which is chosen similarly to $\mathcal{D}_2$, except that the planted subgraph H is now an *arbitrary* graph with average degree k^β (that is, log-density β). Here, the algorithm must be able to detect the second case with high probability regardless of the choice of H.

Finally, we consider the *Dense versus Random* problem, in which we would like to distinguish between $G \sim \mathcal{D}_1$, and an *arbitrary* graph G which contains a k-subgraph H of log-density β.

Observe that for $G \sim \mathcal{D}_1$, a k-subgraph would have expected average degree $kp = kn^{\alpha-1}$. Further, it can be shown that densest k-subgraph in G will have average degree $\max\{kn^{\alpha-1}, 1\}$, w.h.p. (up to a logarithmic factor). Thus if we can solve the distinguishing problem above, its 'distinguishing ratio' would be $\min_\beta(k^\beta / \max\{kn^{\alpha-1}, 1\})$, where β ranges over all values for which we can distinguish (for the corresponding values of k, α). If this is the case for all $\beta > \alpha$, then (as follows from a straightforward calculation), the distinguishing ratio is never more than

$$\begin{aligned}
\frac{k^\alpha}{\max\{kn^{\alpha-1}, 1\}} &= \min\left\{\left(\frac{n}{k}\right)^{1-\alpha}, k^\alpha\right\} \\
&= n^{\alpha(1-\alpha)} \cdot \min\left\{\left(\frac{n^{1-\alpha}}{k}\right)^{1-\alpha}, \left(\frac{k}{n^{1-\alpha}}\right)^{\alpha}\right\} \\
&\leq n^{\alpha(1-\alpha)} \\
&\leq n^{1/4}.
\end{aligned}$$

In this section we will only discuss the Random Planted Model and the Dense versus Random problem, while the intermediate Dense in Random problem is only examined in Section 6.

3.1 The random planted model

One easy way of distinguishing between the two distributions in the Random Planted Model involves looking at the highest degree vertices, or at the pairs of vertices with the largest intersection of neighborhoods. This approach, which is discussed in Section 6 is not only a distinguishing algorithm, but can also identify H in the case of $G \sim \mathcal{D}_2$. However, it is not robust, in the sense that we can easily avoid a detectable contribution to the degrees of vertices of H by resampling the edges between H and $G \setminus H$ with the appropriate probability.

Rather, we examine a different approach, which is to look for constant size subgraphs H' which act as 'witnesses'. If $G \sim \mathcal{D}_1$, we want that w.h.p. G will not have a subgraph isomorphic to H', while if $G \sim \mathcal{D}_2$, w.h.p. G should have such a subgraph. It turns out that whenever $\beta > \alpha$, such an H' can be exists, and thus we can solve the distinguishing problem.

Standard probabilistic analysis (cf. [AS08]) shows that if a graph has log-density greater than r/s (for fixed integers $0 < r < s$) then it is expected to have constant size subgraphs in which the ratio of edges to vertices is $s/(s-r)$, and if the log-density is smaller than r/s, such subgraphs are not likely to exist (i.e., the occurrence of such subgraphs has a threshold behavior). Hence such subgraphs can serve as witnesses when $\alpha < r/s < \beta$.

Observe that in the approach outlined above, r/s is rational, and the size of the witnesses increases as r and s increase. This serves as intuition as to why the statement of Theorem 1.2 involves a rational number r/s, with the running time depending on the value of r.

3.2 Dense versus Random

The random planted model above, though interesting, does not seem to say much about the general DkS problem. In particular, for the Dense versus Random problem, simply looking for the occurrence of subgraphs need not work, because the planted graph could be very dense and yet not have the subgraph we are looking for.

To overcome this problem, we will use a different kind of witness, which will involve special constant-size trees, which we call *templates*. In a *template witness* based on a tree T, we fix a

small set of vertices U in G, and count the number of trees isomorphic to T whose set of leaves is exactly U. The templates are chosen such that a random graph with log-density below a threshold will have a count at most poly-logarithmic for *every* choice of U, while we will show by a counting argument that in any graph (or subgraph) with log-density above the same threshold, there exists a set of vertices U which coincide with the leaves of at least n^ε copies of T (for some constant $\varepsilon > 0$). As noted in Section 2, we may assume *minimum* degree k^β in H as opposed to average degree (this will greatly simplify the counting argument).

As an example, suppose the log-density is $2/3$. In this case, the template T we consider is the tree $K_{1,3}$ (a claw with three leaves). For any triple of vertices U, we count the number of copies of T with U as the set of leaves – in this case this is precisely the number of common neighbors of the vertices in U. In this case, we show that if $G \sim \mathcal{D}_1$, with $\alpha \leq 2/3$, *every* triple of vertices has at most $O(\log n)$ common neighbors. While in the dense case, with $\beta = 2/3 + \varepsilon$, there exists some triple with at least k^ε common neighbors. Since for ranges of parameters of interest $k^\varepsilon = \omega(\log n)$, we have a distinguishing algorithm.

Let us now consider a log-density threshold of r/s (for some relatively prime integers $s > r > 0$). The tree T we will associate with the corresponding template witness will be a caterpillar – a single path called the *backbone* from which other paths, called *hairs*, emerge. In our case, the hairs will all be of length 1. More formally,

Definition 3.1. An (r, s)-*caterpillar* is a tree constructed inductively as follows: Begin with a single vertex as the leftmost node in the backbone. For s steps, do the following: at step i, if the interval $[(i-1)r/s, ir/s]$ contains an integer, add a hair of length 1 to the rightmost vertex in the backbone; otherwise, add an edge to the backbone (increasing its length by 1).

This inductive definition is also useful in deriving an upper bound on the number of (r, s)-caterpillars in $G(n, p)$ (for $p \leq n^{r/s-1}$) with a fixed sequence of 'leaves' (end-points of the hairs) $v_0, v_1, \ldots, v_r$. We do this by bounding the number of candidates for each internal (backbone) vertex, and showing that with high probability, this is at most $O(\log n)$. We begin by bounding the number of candidates for the rightmost backbone vertex in a *prefix* of the (r, s) caterpillar (as per the above inductive construction). For each $t = 1, \ldots, r$, let us write $S_{v_0,\ldots,v_{\lfloor tr/s \rfloor}}(t)$ for the set of such candidates at step t (given the appropriate prefix of leaves). The following claim upper bounds the cardinality of these sets (with high probability). (Recall the notation $\mathrm{fr}(x) = x - \lfloor x \rfloor$.)

Claim 3.2. *In $G(n, p)$, for $p \leq n^{r/s-1}$, for every $t = 1, \ldots, s$ and for any fixed sequence of vertices $U_i = v_0, \ldots, v_{\lfloor tr/s \rfloor}$, for every vertex $v \in V \setminus U_i$ we have*

$$\Pr[v \in S_{v_0,\ldots,v_{\lfloor tr/s \rfloor}}(t)] \leq n^{\mathrm{fr}(tr/s)-1}(1 + o(1)).$$

Intuitively, the claim follows from two simple observations: (a) For any set of vertices $S \subseteq V$ in $G(n, p)$, w.h.p. the neighborhood of S has cardinality at most $pn|S|$ (since the degree of every vertex is tightly concentrated around pn), and (b) for every vertex set S, the expected cardinality of its intersection with the neighborhood of any vertex v is at most $\mathbb{E}[|S \cap \Gamma(v)|] \leq p|S|$. Applying these bounds inductively to the construction of the sets $S(t)$ when $p = n^{r/s-1}$ then implies $|S(t)| \leq n^{\mathrm{fr}(tr/s)}$ for every t.

Proof (sketch). In fact, it suffices to show equality for $p = n^{r/s-1}$ (since for sparser random graphs the probability can only be smaller). More precisely, for this value of p, we show:

$$\Pr[v \in S_{v_0,\ldots,v_{\lfloor tr/s \rfloor}}(t)] = n^{\mathrm{fr}(tr/s)-1}(1 \pm o(1)).$$

We prove the claim by induction. For $i = 1$, it follows by definition of $G(n,p)$: $\Pr[v \in S_{v_0}(1)] = p = n^{r/s-1}$. For $t > 1$, assume the claim holds for $t-1$. If the interval $[(t-1)r/s, tr/s]$ contains an integer (for $1 < t \leq s$ it must be $\lceil (t-1)r/s \rceil$), then $S(t) = S(t-1) \cap \Gamma(v_{\lceil (t-1)r/s \rceil})$. Thus, by definition of $G(n,p)$ and the inductive hypothesis,

$$\begin{aligned}\Pr[v \in S_{v_0,\ldots,v_{\lfloor tr/s \rfloor}}(t)] &= p \cdot \Pr[v \in S_{v_0,\ldots,v_{\lfloor (t-1)r/s \rfloor}}(t-1)] \\ &= n^{r/s-1} n^{\mathrm{fr}((t-1)r/s)-1}(1+o(1)) \\ &= n^{\mathrm{fr}(tr/s)-1}(1 \pm o(1)).\end{aligned}$$

Otherwise, if the interval $[(t-1)r/s, tr/s]$ does not contain an integer, then $S(t) = \Gamma(S(t-1))$. In this case, by the inductive hypothesis, the cardinality of the set $|S(t-1)|$ is tightly concentrated around $n^{\mathrm{fr}((t-1)r/s)}$ (using Chernoff-Hoeffding bounds). If we condition on the choice of all $S(t')$ for $t' < t$, and $S(t-1)$ has (approximately) the above cardinality, then for every v not appearing in the previous sets, we have

$$\begin{aligned}\Pr[v \in S_{v_0,\ldots,v_{\lfloor tr/s \rfloor}}(t)] &= \Pr[\exists u \in S_{v_0,\ldots,v_{\lfloor tr/s \rfloor}}(t-1) : (u,v) \in E] \\ &= 1 - (1-p)^{|S(t-1)|} \\ &= p|S(t-1)|(1-o(1)) && \text{since } p|S(t-1)| = o(1) \\ &= n^{r/s-1} n^{\mathrm{fr}((t-1)r/s)}(1 \pm o(1)) \\ &= n^{\mathrm{fr}(tr/s)-1}(1 \pm o(1)).\end{aligned}$$

Note that a more careful analysis need also bound the number of vertices participating in $S(t) \cap S(t')$ for all $t' < t$. Further, even in this case, tight concentration assumed above is only achieved when the expected size of the set is $n^{\Omega(1)}$. However, this is guaranteed by the inductive hypothesis, assuming r and s are relatively prime. □

Now by symmetry, the same bounds can be given when constructing the candidate sets in the opposite direction, from right to left (note the symmetry of the structure). Thus, once all the leaves are fixed, every candidate for an internal vertex can be described, for some $t \in [1, s-1]$, as the rightmost backbone vertex in the tth prefix, as well as the leftmost backbone vertex in the $(s-t)$th prefix starting from the right. By Claim 3.2, the probability of this event is at most

$$n^{\mathrm{fr}(tr/s)-1} n^{\mathrm{fr}((s-t)r/s)-1}(1+o(1)) = n^{-1}(1+o(1)).$$

Thus, since the (r,s)-caterpillar has $s-r$ internal vertices and $r+1$ leaves, it follows by standard probabilistic arguments that, for some universal constant $C > 0$, the probability that total number of caterpillars for any sequence of leaves exceeds $(\log n)^{s-r}$ is at most $(s-r)n^{r+1}n^{-C \log\log n}$, which is $o(1)$ for any constants r, s.

Now let us consider the number of (r,s)-caterpillars with a fixed set of leaves in a k-subgraph H with minimum degree at least $d = k^{(r+\varepsilon)/s}$. Ignoring low-order terms (which would account for repeated leaves), the number of (r,s)-caterpillars in H (double counting each caterpillar to account for constructing it inductively once from each direction) is at least kd^s (since it is a tree with s edges), whereas the number of possible sequences of leaves is at most k^{r+1}. Thus, the number of (r,s) caterpillars in H corresponding to the average sequence of $r+1$ H-vertices is at least $kd^s/k^{r+1} = k^{r+\varepsilon}/k^r = k^\varepsilon$. Note that the parameters for the high probability success of the dense-versus-random distinguishing algorithm are the same as for the random planted model, giving, as before, an distinguishing ratio of $\tilde{O}(n^{1/4})$ in the worst case.

4 An LP based algorithm for arbitrary graphs

We now give a general algorithm for DkS inspired by the distinguishing algorithm in the Dense vs Random setting. For a graph G with maximum degree $D = n^{r/s}$, we will use the (r, s)-caterpillar template, and keep track of sets $S(t)$ as before. We then fix the leaves one by one, while maintaining suitable bounds on $S(t)$.

Let us start by describing the LP relaxation.[2] Taking into account the simplifications from Section 2, we define a hierarchy of LPs which is satisfied by a graph which contains a subgraph of size at most k with minimum degree at least d. This hierarchy is at most as strong as the Lovász-Schrijver LP hierarchy based on the usual LP relaxation (and is possibly weaker). Specifically, for all integers $t \geq 1$, we define $\mathbf{DkS\text{-}LP}_t(G, k, d)$ to be the set of n-dimensional vectors $(y_1, \ldots y_n)$ satisfying:

$$\sum_{i \in V} y_i \leq k, \qquad \text{and} \tag{1}$$

$$\exists \{y_{ij} \mid i, j \in V\} \qquad \text{s.t.}$$

$$\forall i \in V \qquad \sum_{j \in \Gamma(i)} y_{ij} \geq d y_i \tag{2}$$

$$\forall i, j \in V \qquad y_{ij} = y_{ji} \tag{3}$$

$$\forall i, j \in V \qquad 0 \leq y_{ij} \leq y_i \leq 1 \tag{4}$$

$$\text{if } t > 1, \forall i \in V \text{s.t. } y_i \neq 0 \qquad \{y_{i1}/y_i, \ldots, y_{in}/y_i\} \in \mathbf{DkS\text{-}LP}_{t-1}(G, k, d) \tag{5}$$

Given an LP solution $\{y_i\}$, we write $\mathrm{LP}_{\{y_i\}}(S) = \sum_{i \in S} y_i$. When the solution is clear from context, we denote the same by $\mathrm{LP}(S)$. We call this the *LP-value* of S. When the level in the hierarchy will not be important, we will simply write DkS-LP instead of DkS-LP$_t$. A standard argument shows that a feasible solution to DkS-LP$_t(G, k, d)$ (along with all the recursively defined solutions implied by constraint (5)) can be found in time $n^{O(t)}$. For completeness, we illustrate this in Appendix B

Informally, we can think of the LP as giving a distribution over subsets of V, with y_i being the probability that i is in a subset. Similarly y_{ij} can be thought of as the probability that both i, j are 'picked'. We can now think of the solution $\{y_{ij}/y_i : 1 \leq j \leq n\}$ as a distribution over subsets, conditioned on the event that i is picked.

Algorithm outline. The execution of the algorithm follows the construction of an (r, s)-caterpillar. We perform s steps, throughout maintaining a subset $S(t)$ of the vertices. For each t, we perform either a 'backbone step' or a 'hair step' (which we will describe shortly). In each of these steps, we will either find a dense subgraph, or extend an inductive argument that will give a lower bound on the ratio $\mathrm{LP}(S(t))/|S(t)|$. Finally, we show that if none of the steps finds a dense subgraph, then we reach a contradiction in the form of a violated LP constraint, namely $LP(S(s)) > |S(s)|$.

4.1 The Algorithm

Let us now describe the algorithm in detail. The algorithm will take two kinds of steps, *backbone* and *hair*, corresponding to the two types of caterpillar edges. While these steps differ in the

[2]The entire algorithm can be executed without solving an LP, by performing an exhaustive search for the best set of leaves, with running time comparable to that of solving the LP. We will elaborate on this later.

updates they make, both use the same procedure to search locally for a dense subgraph starting with a current candidate-set. Let us now describe this procedure.

DkS-Local(S, k)

- Consider the bipartite subgraph induced on $(S, \Gamma(S))$.
- For all $k' = 1, \ldots, k$, do the following:
  - Let $T_{k'}$ be the set of k' vertices in $\Gamma(S)$ with the highest degree (into S).
  - Take the $\min\{k', |S|\}$ vertices in S with the most neighbors in $T_{k'}$, and let $H_{k'}(S)$ be the bipartite subgraph induced on this set and $T_{k'}$.
- Output the subgraph $H_{k'}(S)$ with the largest average degree.

We will analyze separately the performance of this procedure in the context of a leaf-step and that of a hair-step. We begin by relating the performance of this procedure to an LP solution.

Claim 4.1. *Given a set of vertices $S \subseteq V$, and an LP solution $\{y_i\} \in$ DkS-LP(G, k, d), let $k' = \lceil \text{LP}(\Gamma(S)) \rceil$. Then* DkS-Local$(S, k)$ *outputs a subgraph with average degree at least*

$$\frac{1}{\max\{|S|, k'\}} \cdot \sum_{j \in \Gamma(S)} y_j |\Gamma(j) \cap S|.$$

Proof. Note that by constraint (1), $k' = \lceil \text{LP}(\Gamma(S)) \rceil \leq k$. Then in Procedure DkS-Local, the vertices in $T_{k'}$ must have at least $\sum_{j \in \Gamma(S)} y_j |\Gamma(j) \cap S|$ edges to S: indeed, the summation $\sum y_j |\Gamma(j) \cap S|$ can be achieved by taking $\sum_{j \in T_{k'}} 1 \cdot |\Gamma(j) \cap S|$ and moving some of the weight from vertices in $T_{k'}$ to lower-degree (w.r.t. S) vertices (and perhaps throwing some of the weight away). After choosing the $\min\{k', |S|\}$ vertices in S with highest degree, the remaining subgraph $H_{k'}(S)$ has average degree at least

$$\frac{\min\{k', |S|\}}{|S|} \cdot \frac{1}{k'} \cdot \sum_{j \in \Gamma(S)} y_j |\Gamma(j) \cap S|.$$

This proves the claim. □

The backbone step in the algorithm first performs DkS-Local on the current S, and then sets S to be $\Gamma(S)$. The following lemma gives a way to inductively maintain a lower bound on $\text{LP}(S(t))/|S(t)|$ assuming DkS-Local does not find a sufficiently dense subgraph.

Lemma 4.2. *Given $S \subseteq V$, and an LP solution $\{y_i\}$ for* DkS-LP(G, k, d)*: for any $\rho \geq 1$ such that $\text{LP}(S)/|S| \geq \rho/d$, either* DkS-Local$(S, k)$ *outputs a subgraph with average degree at least ρ or we have*

$$\text{LP}(\Gamma(S)) \geq \frac{d\text{LP}(S)}{\rho}.$$

Proof. By the LP constraints (4) and (2), we have

$$\sum_{j \in \Gamma(S)} y_j |\Gamma(j) \cap S| \geq \sum_{j \in \Gamma(S)} \sum_{i \in \Gamma(j) \cap S} y_{ij} = \sum_{i \in S} \sum_{j \in \Gamma(i)} y_{ij} \geq d\text{LP}(S).$$

By Claim 4.1, Dks-Local(S,k) outputs a subgraph with average degree at least $d\text{LP}(S)/\max\{|S|,k'\}$, where $k' = \text{LP}(\Gamma(S))$ (note that we are ignoring some roundoff error which will be negligible in the context of the algorithm).

If $k' \leq |S|$, then we are done, since by our assumption, $d\text{LP}(S)/|S| \geq \rho$. Now suppose $k' \geq |S|$. The output graph has average degree at least $d\text{LP}(S)/k'$. If this is at least ρ, we are done. If not, $k' \geq d\text{LP}(S)/\rho$, and since $k' = \text{LP}(\Gamma(S))$, we get the desired result. □

Let us now consider a hair step. In this case, the algorithm performs DkS-Local on the current set, and then picks a vertex $j \in V$ to act as a "leaf". The new S is then set to equal $S \cap \Gamma(j)$. The following lemmas prove that either DkS-Local finds a sufficiently dense subgraph, or we can pick j so as to inductively maintain certain bounds. Let us first prove a simple averaging lemma.

Lemma 4.3. *Let x_j, $(1 \leq j \leq n)$ be reals in $[0,1]$, with $\sum_j x_j \leq k$. Let P_j and Q_j be some non-negative real numbers such that for some $P, Q > 0$,*

$$\sum_j x_j P_j \geq P, \text{ and } \sum_j x_j Q_j \leq Q. \tag{6}$$

Then there exists an j such that $P_j \geq P/(2k)$ and $P_j/Q_j \geq P/(2Q)$.

Proof. By our assumption $\sum_j x_j\left(P_j - \frac{P}{2k}\right) \geq P - \frac{P}{2} = \frac{P}{2}$. Thus from (6), it follows that there exists an j such that $x_j > 0$ and

$$P_j - \frac{P}{2k} \geq \frac{P}{2Q} \cdot Q_j.$$

This choice of j clearly satisfies the required properties. □

Lemma 4.4. *Let $S \subseteq V$, and let $\{y_i\} \in \text{DkS-LP}(G,k,d)$ be an LP solution (for which there exist corresponding $\{y_{ij}\}$). Then for any $\rho \geq 1$, either* DkS-Local(S,k) *outputs a ρ-dense subgraph, or there exists some vertex $j \in G$, such that $y_j > 0$, and*

$$\text{LP}_{\{y_{ij}/y_j | i \in V\}}(S \cap \Gamma(j)) \geq \frac{d \cdot \text{LP}_{\{y_i\}}(S)}{2k}, \text{ and}$$

$$\text{LP}_{\{y_{ij}/y_j | i \in V\}}(S \cap \Gamma(j))/|S \cap \Gamma(j)| \geq \frac{d \cdot \text{LP}_{\{y_i\}}(S)}{2\rho \cdot \max\{k, |S|\}}.$$

Proof. By constraints (2) of the LP we have

$$\sum_{j \in \Gamma(S)} y_j LP_{\{y_{ij}/y_j | i \in V\}}(S \cap \Gamma(j)) = \sum_{j \in \Gamma(S)} \sum_{i \in \Gamma(j) \cap S} y_{ij} = \sum_{i \in S} \sum_{j \in \Gamma(i)} y_{ij} \geq d\text{LP}_{\{y_i\}}(S). \tag{7}$$

From Claim 4.1, it follows that if the subgraph found by DkS-Local has average degree less than ρ, we must have $\rho > \sum_{j \in \Gamma(S)} y_j |\Gamma(j) \cap S| / \max\{|S|, k'\}$, or in other words

$$\sum_{j \in \Gamma(S)} y_j |\Gamma(j) \cap S| \leq \rho \max\{|S|, k'\} \leq \rho \max\{|S|, k\}. \tag{8}$$

Thus the lemma follows immediately from Lemma 4.3 and equations (7) and (8). □

We now formally describe the algorithm. It takes as input a graph G, a parameter k, and $\{y_i\}$, a solution to DkS-LP$_{r+2}(G, k, d)$. Throughout, a set $S \subseteq V$, and an LP solution $\{y_i\}$ are maintained.

DkS-Cat$_{r,s}(G, k, \{y_i\})$

- Let $S_0 = V$.
- For all $t = 1, \ldots, s$, do the following:
  - For $t > 1$, let H_t be the output of Procedure DkS-Local(S_{t-1}, k).
  - If the interval $[(t-1)r/s, tr/s]$ contains an integer, perform a **hair step**: Choose some vertex j_t as in Lemma 4.4 (or for $t = 1$, choose any j_1 such that $y_{j_1} > 0$), and
    - Let $S_t = S_{t-1} \cap \Gamma(j_t)$.
    - Replace the LP solution $\{y_i\}$ with $\{y_{ij_t}/y_{j_t} \mid i \in V\}$.
  - Otherwise, perform a **backbone step**: Let $S_t = \Gamma(S_{t-1})$.
- Output the subgraph H_t with the highest average degree.

Note that since the "conditioning" (replacing y_i's by y_{ij}/y_j) in the hair steps is only performed $r + 1$ times, then by constraint (5), at every step of the algorithm $\{y_i\}$ satisfies DkS-LP(G, k, d).

A combinatorial algorithm. Note that the only time the algorithm uses the LP values is in choosing the leaves. Thus, even in the absence of an LP solution, the algorithm can be run by trying all possible sequences of leaves (the analysis will still work by replacing the LP solution with the optimum $0 - 1$ solution). While this would take time $O(n^{r+1})$ as opposed to linear time (for the LP-based rounding algorithm), this is comparable to the time needed to solve the LP. An interesting open question is if it is possible to avoid the dependence on r, the number of leaves.

4.2 Performance guarantee

The analysis is quite straightforward. We follow the various steps, and each time apply either Lemma 4.2 or Lemma 4.4, as appropriate. Our main result is the following:

Theorem 4.5. *Let $s > r > 0$ be relatively prime integers, let G be an undirected (bipartite) graph with maximum degree $\leq D = n^{r/s}$, let $\{y_i\} \in$ DkS-LP$_{r+1}(G, k, d)$. and define $\gamma = \max\{Dk/n, 1\}$. Then if d' is the average degree of the subgraph found by* DkS-Cat$_{r,s}(G, k, \{y_i\})$*, we have*

$$\max\{d', \gamma\} = \Omega(d/D^{(s-r)/s}).$$

Note that when the log-density α of G is not rational, we can choose rational $\alpha \leq r/s \leq \alpha + \varepsilon$ for any small $\varepsilon > 0$. We then still appeal to Theorem 4.5 as before, though the greedy algorithm might only return a subgraph of average degree $\gamma' > \gamma/n^{\varepsilon}$. Thus, the loss in the approximation ratio is at most n^{ε}. A fairly straightforward calculation shows that this implies a $O(n^{1/4+\varepsilon})$-approximation in $O(n^{O(1/\varepsilon)})$ time for all $\varepsilon > 0$ (including $\varepsilon = 1/\log n$).

Before going into the details of the proof, let us note the similarities between the algorithm and the random models discussed earlier. Recall that in the random case, the sets $S(t)$ (corresponding to S_t in the algorithm) had cardinality tightly concentrated around $n^{\mathrm{fr}(tr/s)}$. Similarly here, if we assume that $k = n/D (= D^{(s-r)/r})$, and that d (the density of the subgraph implied by the LP) is at least $\rho k^{r/s}$ (for some $\rho \geq 1$), then we show (see Corollary 4.7) that if until step t the algorithm has not found an $\Omega(\rho)$-dense subgraph, then the current candidate set satisfies

$$\frac{\mathrm{LP}(S_t)}{|S_t|} > \rho^{-\mathrm{fr}(tr/s)\cdot s/r} \left(\frac{k}{n}\right)^{\mathrm{fr}(tr/s)} = \left(\frac{1}{D}\right)^{\mathrm{fr}(tr/s)},$$

which will yield a contradiction after the final step (when $t = s$).

One difficulty is that we avoid making the assumption that $kD = n$ (which is possible, but would incur a $O(\sqrt{\log n})$ loss in the approximation guarantee). Instead, we use the fact that the greedy algorithm finds a k-subgraph with average degree $\gamma \geq \max\{1, Dk/n\}$. Specifically, we show that at step t of the algorithm, either a subgraph with average degree $\Omega(\rho)$ has already been found, or the greedy algorithm gives the desired approximation (i.e. $\gamma \geq \rho$), or we have the desired lower bounds on $\mathrm{LP}(S_t)$ and $\mathrm{LP}(S_t)/|S_t|$.

Notation. In what follows, we let $\rho = d/(2D^{(s-r)/s})$ denote the desired average degree of the output subgraph (up to a constant factor). We also write $L_t = \lfloor tr/s \rfloor$. Note that the number of hair steps up to and including step t is precisely $L_t + 1$.

We now state the main technical lemma.

Lemma 4.6. *Let $s > r > 0$ be relatively prime integers, let G be an undirected (bipartite) graph with maximum degree at most $D = n^{r/s}$, and let $\{y_i\}$ be a solution to* DkS-LP$_{r+1}(G, k, d)$. *Let $\gamma = \max\{Dk/n, 1\}$. For $t = 1, \ldots, s$, let d'_t be the average degree of the densest of the subgraphs found by* DkS-Cat$_{r,s}(G, k, \{y_i\})$ *up through step t. Then either*

$$\max\{d'_t, \gamma\} = \Omega(\rho),$$

or we have

$$\begin{aligned} \mathrm{LP}(S_t) &\geq \frac{d^t}{2^{L_t}\rho^{t-L_t-1}k^{L_t}}, \text{ and} \\ \frac{\mathrm{LP}(S_t)}{|S_t|} &\geq \frac{d^t}{2^{L_t}\gamma\rho^{t-1}D^{t-L_t}}. \end{aligned} \tag{9}$$

The following simple corollary immediately implies Theorem 4.5 (by contradiction) when we take $t = s$.

Corollary 4.7. *In the notation of Lemma 4.6, either* $\max\{d'_t, \gamma\} = \Omega(d/D^{(s-r)/s})$, *or we have*

$$\frac{\mathrm{LP}(S_t)}{|S_t|} \geq \frac{2^{t-\lfloor tr/s \rfloor}}{D^{\mathrm{fr}(tr/s)}}.$$

Proof.

$$\begin{aligned}
\frac{\mathrm{LP}(S_t)}{|S_t|} &\geq \frac{d^t}{2^{L_t}\gamma\rho^{t-1}D^{t-L_t}} && \text{By Lemma 4.6}\\
&> \frac{d^t}{2^{L_t}\rho^t D^{t-L_t}} && \rho > \gamma\\
&= \frac{d^t}{2^{L_t}D^{t-L_t}} \cdot \frac{(2D^{(s-r)/s})^t}{d^t} && \text{definition of } \rho\\
&= \frac{2^{t-L_t}}{D^{t-L_t-t(s-r)/s}} = \frac{2^{t-L_t}}{D^{tr/s-L_t}}.
\end{aligned}$$

□

Let us now proceed to the proof of Lemma 4.6.

Proof of Lemma 4.6. We prove by induction that if the algorithm does not find an $\Omega(\rho)$ dense subgraphs in steps 1 through t, the lower bounds (9) hold. Assume that $\rho > \gamma(\geq 1)$ (otherwise we are done).

For $t = 1$, the bounds hold trivially: If j_1 is any vertex for which $y_{j_1} > 0$, then we have $\mathrm{LP}_{\{y_{ij_1}/y_{j_1}\}}(S_1) \geq d$ (by constraint (2)) and so $\mathrm{LP}(S_1)/|S_1| \geq d/|\Gamma(j_1)| \geq d/D(\geq d/(\gamma D))$, which is exactly what we need.

Now assume the lemma holds for some $1 \leq t \leq s-1$. We will show it for $t+1$, considering separately backbone and hair steps.

First, suppose $t+1$ is a backbone step. Then the interval $[tr/s, (t+1)r/s]$ does not contain an integer, i.e. $tr/s < L_t + (s-r)/s$. By Lemma 4.2, if $LP(S_t)/|S_t| \geq \rho/d$ then either the procedure DkS-Local(S_t, k) produces a ρ-dense subgraph, or we have $\mathrm{LP}(S_{t+1}) = \mathrm{LP}(\Gamma(S_t)) \geq d\mathrm{LP}(S_t)/\rho$. In that case, since $|S_{t+1}| = |\Gamma(S_t)| \leq D|S_t|$, the claim follows immediately from the inductive hypothesis. Thus it suffices to show that indeed $LP(S_t)/|S_t| \geq \rho/d$. This follows from Corollary 4.7, which gives

$$\frac{\mathrm{LP}(S_t)}{|S_t|} \geq \frac{2^{t-L_t}}{D^{tr/s-L_t}} > \frac{2^{t-L_t}}{D^{(s-r)/s}} = \frac{2^{t+1-L_t}\rho}{d}.$$

Now, suppose $t+1$ is a hair step. Then the interval $[tr/s, (t+1)r/s]$ does contain an integer, i.e. $tr/s \geq L_t + (s-r)/s$. Assuming Procedure DkS-Local(S_t, k) does not return a subgraph with average degree at least ρ, by Lemma 4.4, there is some choice of vertex j_{t+1} such that

$$\mathrm{LP}_{\{y_{ij_{t+1}}/y_{j_{t+1}}|i\in V\}}(S_{t+1}) = \mathrm{LP}_{\{y_{ij_{t+1}}/y_{j_{t+1}}|i\in V\}}(S_t \cap \Gamma(j_{t+1})) \geq \frac{d \cdot \mathrm{LP}_{\{y_i\}}(S_t)}{2k}, \text{ and} \tag{10}$$

$$\mathrm{LP}_{\{y_{ij_{t+1}}/y_{j_{t+1}}|i\in V\}}(S_{t+1})/|S_{t+1}| \geq \frac{d \cdot \mathrm{LP}_{\{y_i\}}(S_t)}{2\rho \cdot \max\{k, |S_t|\}}. \tag{11}$$

Thus, the required lower bound $\mathrm{LP}(S_{t+1})$ follows from (10) and the inductive hypothesis. If $|S_t| \geq k$, the bound on $\mathrm{LP}(S_{t+1})/|S_{t+1}|$ similarly follows from (11) and the inductive hypothesis. Thus, it

remains to show the required bound when $|S_t| < k$. In that case, (11) gives

$$\begin{aligned}
\mathrm{LP}(S_{t+1})/|S_{t+1}| &\geq \frac{d\mathrm{LP}(S_t)}{2\rho k} \\
&\geq \frac{d^{t+1}}{2^{L_t+1}\rho^{t-L_t}k^{L_t+1}} && \text{inductive hypothesis} \\
&> \frac{d^{t+1}}{2^{L_t+1}\rho^{t}} \cdot \frac{\gamma^{L_t}}{k^{L_t+1}} && \text{since } \rho > \gamma \\
&\geq \frac{d^{t+1}}{2^{L_t+1}\rho^{t}} \cdot \frac{D^{L_t+1}}{\gamma n^{L_t+1}} && \text{since } \gamma \geq kD/n \\
&\geq \frac{d^{t+1}}{2^{L_t+1}\rho^{t}} \cdot \frac{D^{L_t+1}}{\gamma n^{(t+1)r/s}} && \text{since } tr/s \geq L_t + (s-r)/s \\
&\geq \frac{d^{t+1}}{2^{L_t+1}\rho^{t}} \cdot \frac{1}{\gamma D^{t+1-(L_t+1)}}. && \text{since } D \geq n^{r/s}
\end{aligned}$$

Since $L_{t+1} = L_t + 1$, the bound follows. This concludes the proof. □

5 An $n^{(1/4-\varepsilon)}$-approximation with 'mildly' exponential run time

We will now consider an extension of our approach to the case when the log-density of the subgraph H is (slightly) less than the log-density of the host graph (a crucial case if one wants to go beyond $n^{1/4}$-approximations). This is done at the expense of running time – we obtain a modification of our caterpillar-based algorithm, which yields an approximation ratio of $O(n^{(1-\varepsilon)/4})$ approximation which runs in time $2^{n^{O(\varepsilon)}}$. The main modification is that for each leaf, rather than picking an individual vertex, we will pick a cluster of roughly $O(n^{\varepsilon})$ vertices (which is responsible for the increased running time). The cluster will be used similarly to a single leaf vertex: rather than intersecting the current set with the neighborhood of a single vertex, we will intersect the current set with the neighborhood of the cluster (i.e., the union of all neighborhoods of vertices in the cluster).

Motivation. We now describe briefly why we could expect such a procedure to work (and why we *need* to look at sets of size roughly n^{ε}). Suppose we are given a random graph G with log-density ρ. Let r/s denote a rational number roughly equal to $\rho + \delta$ (for some small constant δ). Suppose we call an $(r+1)$ tuple of leaves 'special' if there is an (r, s) caterpillar 'supported' on it, in the sense of Section 3.2. Since $r/s > \rho$, most $(r+1)$-tuples of vertices are not special.

The crucial observation is the following: say we pick a set S of vertices, and ask how many $(r+1)$ tuples from S are special. This number turns out to be 'large' (appropriately defined later) roughly iff $|S| > n^{\delta}$. Now suppose we had planted a random graph H on k vertices and log-density $\rho - \varepsilon$ in G. Further suppose δ is such that $k^{\varepsilon+\delta} \ll n^{\delta}$ (since k is much smaller than n, we can always choose this so, by setting δ to be a constant multiple of ε). By the above claim, sets in H of size $k^{\varepsilon+\delta}$ would have a 'large' number of special tuples (since $r/s =$ (log-density of H) $+ \varepsilon + \delta$). But this number, by choice is much smaller than n^{δ}, thus if there was no H planted, sets of size $k^{\varepsilon+\delta}$ would not have many special tuples!

This gives a distinguishing algorithm which runs in time roughly $\binom{n}{k^{\varepsilon+\delta}}$. Let us now develop the algorithm in detail, for arbitrary graphs.

To simplify the presentation, we will not describe the LP setting here, but only consider the equivalent of the combinatorial algorithm described earlier ("guessing the leaves"). Due to the already large running time, this will make little difference in terms of efficiency. Before we introduce the algorithm and provide a sketch of the analysis, let us state the main result of this section:

Theorem 5.1. *For every $\varepsilon > 0$, there is a randomized $O(2^{n^{6\varepsilon}})$-time algorithm which for every graph G with high probability finds a k-subgraph whose average degree is within $\tilde{O}(n^{(1-\varepsilon)/4})$ of the optimum.*

Theorem 5.1 is an immediate corollary of the performance guarantee of the modified algorithm, which is as follows:

Theorem 5.2. *For every $0 < \varepsilon < \frac{1}{2}$, and every $0 < \beta < 1$, there is a randomized algorithm which for every instance of Dense k-Subgraph with $k = n^\beta$, finds a k-subgraph whose average degree is within $\tilde{O}(n^{\beta(1-\beta)(1-\varepsilon)})$ of the optimum (with high probability). Moreover, this algorithm runs in time at most $2^{\tilde{O}(n^{2\beta\varepsilon/(2\beta\varepsilon+(1-\beta))})}$.*

Let us see that Theorem 5.1 follows easily:

Proof of Theorem 5.1. For $\varepsilon > 1/6$, we can check every subgraph in time $2^n < 2^{n^{6\varepsilon}}$. Thus, we may assume that $\varepsilon \leq 1/6$. Let $k = n^\beta$, and consider separately two cases. First, suppose that $\beta \leq 5/7$. Note that the algorithm referred to in Theorem 5.2 always gives an $\tilde{O}(n^{(1-\varepsilon)/4})$-approximation (regardless of β). Thus we only need to bound the running time. For $\beta \leq 5/7$, the expression in the exponent is

$$n^{2\beta\varepsilon/(2\beta\varepsilon+(1-\beta))} \leq n^{2\beta\varepsilon/(1-\beta)} \leq n^{2(5/7)\varepsilon/(2/7)} = n^{5\varepsilon}.$$

Now consider $\beta > 5/7$. Using the reduction in Appendix A.2, we may assume that $D = n^\alpha$ for $\alpha < n^{2/7}$. Algorithm DkS-Cat described in Section 4 (for $r/s \approx \alpha$) already gives (in quasi-polynomial time) up to a constant factor an approximation ratio of at most

$$D^{1-\alpha} = n^{\alpha(1-\alpha)} < n^{10/49} < n^{(1-1/6)/4} \leq n^{(1-\varepsilon)/4}.$$

□

To simplify the analysis, we may perform additional preprocessing steps which will constrain the range of parameters:

Lemma 5.3. *For the purposes of finding an $\tilde{O}(n^{(1-\varepsilon)/4})$-approximation for DkS, we may assume that the maximum degree of G is $D = n/k$, and that moreover, for α s.t. $D = n^\alpha$, the minimum degree in H is $d = k^{\alpha(1-\delta)}$ for some $0 \leq \delta \leq \varepsilon$.*

Proof (sketch). Let β be s.t. $k = n^\beta$. Then perform the following steps:

1. If $d > k^{1-\beta}$, prune the edges of G such that every edge is retained independently with probability $\left(k^{1-\beta}/d\right)$.
2. Now let D' be the $k/2$-largest degree in the pruned graph.

- If $D'k < n$, add to the current set of edges the edges of the random graph $G(n, 1/k)$. The new edges cannot make more then a negligible contribution to the densest k-subgraph.
- Otherwise, if $D'k > n$, prune as in Appendix A.2.

Step 1 already guarantees that $d \leq k^{1-\beta}$, and step 2 guarantees that the $k/2$-largest degree in the final graph is approximately n/k. By Lemma A.1, w.l.o.g. we can ignore the first $k/2$ vertices. It only remains to check that d is not too small (not less than $k^{(1-\beta)(1-\varepsilon)}$). Arguing as in Appendix A.2, we may assume that if $D'k > n$ then in the final graph we have $d > n^{(1-\varepsilon)/4} = k^{(1-\varepsilon)/(4\beta)} \geq k^{(1-\beta)(1-\varepsilon)}$. □

In the modified algorithm, the backbone step and its analysis is the same as before. The main difference, as we mentioned, is in the leaf step. The following lemma (compare to Lemma 4.4) gives the guarantee of a leaf step when the current set is intersected with the neighborhood of a cluster as opposed to the neighborhood of a single vertex.

Lemma 5.4. *Let $S \subseteq V$, and let $0 < C < k/2$. Then for any $\rho \geq 1$, either* DkS-Local(S, k) *outputs a ρ-dense subgraph, or there exists some cluster of C vertices $J \subset V(H)$, such that either we can extract a ρ-dense k-subgraph from the bipartite subgraph induced on $(J, S \cap \Gamma(J))$ in linear time, or we have*

$$|S \cap \Gamma(J) \cap H| \geq \Omega\left(\frac{dC \cdot |S \cap H|}{\rho k}\right), \text{ and}$$

$$\frac{|S \cap \Gamma(J) \cap H|}{|S \cap \Gamma(J)|} \geq \Omega\left(\frac{d \cdot |S \cap H|}{\rho^2 \cdot \max\{k, |S|\}}\right).$$

Proof (sketch). The proof follows the same lines as that of Lemma 4.4. This gives similar bounds with respect to $\sum_{j \in J} |\Gamma(j) \cap S \cap H|$. An extra ρ factor is lost since the size of the set $\Gamma(J) \cap S \cap H$ may be significantly less than the number of incoming edges from J. However, if it is more than a factor ρ less, then together with J it forms a ρ-dense subgraph (and so a matching guarantee can be found by greedily picking highest J-degree vertices in S). □

We now present the modified algorithm. It takes as input a graph G with maximum degree at most n/k in which the densest k-subgraph has minimum degree in the range $(k^{(1-\beta)(1-\varepsilon)}, k^{(1-\beta)})$ where β is such that $k = n^{\beta}$. These assumptions are valid by Lemma 5.3.

DkS-Exp$_\varepsilon(G, k)$

- Let $\alpha = 1 - \beta$ (so $D = n^\alpha$). Let $\alpha' = \alpha + 2\beta\varepsilon$, and let $0 < r < s < \log n$ be integers such that $|r/s - \alpha'| \leq 1/\log n$. Let $C = n^{2\beta\varepsilon/(2\beta\varepsilon+\alpha)}$. Let $S_0 = V$.
- For all $t = 1, \ldots, s$, do the following:
  - For $t > 1$, let H_t be the output of Procedure DkS-Local(S_{t-1}, k).
  - If the interval $[(t-1)r/s, tr/s]$ contains an integer, perform a **hair step**: "Guess" a set J_t of C vertices and
    - Let $S_t = S_{t-1} \cap \Gamma(J_t)$.
    - Let H'_t be the output of Procedure DkS-Local(J_t, k) run on the subgraph of G induced on $J_t \cup S_t$.
  - Otherwise, perform a **backbone step**: Let $S_t = \Gamma(S_{t-1})$.
- Output the subgraph H_t or H'_t with the highest average degree.

Note that guessing r clusters of size C correctly (such that they all satisfy Lemma 5.4) takes time at most $\binom{n}{C}^r = 2^{O(C \log n)}$, as required by Theorem 5.2. The approximation guarantee is verified by showing that the algorithm finds a subgraph with density $\rho = \Omega(d/k^{\alpha(1-\varepsilon)})$. The proof runs along the same lines as Lemma 4.6, using a combination of Lemma 4.2 and Lemma 5.4 to prove the following lower bounds inductively (assuming no ρ-dense subgraph is found):

$$|S_t \cap H| > \frac{d^t C^{L_t+1}}{\rho^t k^{L_t}}, \text{ and}$$

$$\frac{|S_t \cap H|}{|S_t|} > \frac{d^t}{\rho^{t+L_t} D^{t-L_t}}.$$

In particular, a simple calculation shows that the second bound implies

$$\frac{|S_t \cap H|}{|S_t|} > \frac{1}{D^{tr/s-L_t}}$$

which for $t = s$ is a contradiction (indicating that a ρ-dense subgraph must have been found).

6 Going beyond log-density in random models

We now show that for a certain range of parameters, we can do better than the log-density bound in Section 3 and later. Specifically, we show this for $D < \sqrt{n}$ (by the simplification in Section A.2, we can use this interchangeably with the condition $k > \sqrt{n}$, which we will in this section).

6.1 Improvements in the Random Planted Model

In the easiest model, the Random Planted Model, if $D \leq \sqrt{n}$ (that is, $p < 1/\sqrt{n}$), then H is simply composed of the k vertices of highest degree in G whenever the expected average degree in H is $\gg \sqrt{\log n}\sqrt{D}$ (this is because with high probability, the minimum degree in H is considerably larger

than the standard deviation of degrees in G). This is an improvement over the log-density bound in this range which in the worst case gives a distinguishing ratio of $D^{1-\alpha}$. A similar argument using the standard deviation of intersections of neighborhoods gives an $n^{1/4}$ distinguishing ratio even for larger D, however for $D \geq \sqrt{n}$ it does not surpass the log-density barrier.

6.2 Improvements for the Dense in Random problem

We now show that in the Dense in Random problem, a simple eigenvalue approach does better than the log-density bound of Section 3.2 for certain ranges of k and the average degree D. Suppose G is a random graph, with log-density ρ. In this case, the results of Section 3.2 allow us to distinguish random G from graphs which have a planted k-subgraph of density $\gg k^\rho$.

Let us now consider the second eigenvalue of the input graph. If it were a random graph with log-density ρ, Füredi and Kómlos [FK81] show that $\lambda_2 \leq O(n^{\rho/2})$, w.h.p. Now suppose G contains a subgraph H on k vertices with min degree $d \gg n^{\rho/2} + \frac{kn^\rho}{n}$. Consider the vector x (in $\mathbb{R}^n$, with coordinates indexed by vertices of G) defined by

$$x_i = \begin{cases} 1, \text{ if } i \in V(H) \\ -\frac{k}{n-k} \text{ otherwise} \end{cases}$$

Note that x is orthogonal to the all-ones vector.[3] Further, if A_G is the adjacency matrix of G, we have that

$$\begin{aligned} \frac{x^T A_G x}{x^T x} &= \frac{\sum_{(i,j)\in E(G)} x_i x_j}{k + \frac{k^2}{n-k}} \\ &\geq \frac{kd - kn^\rho \cdot \frac{k}{n-k}}{2k} \geq \frac{d}{2} - \frac{kn^\rho}{n} \end{aligned}$$

Thus if $d \gg n^{\rho/2} + \frac{kn^\rho}{n}$, we have $\lambda_2 \gg n^{\rho/2}$, and thus we can distinguish between the two cases. We can now compare this threshold for d to the one obtained by a log-density consideration (k^ρ). Thus when $k \gg \sqrt{n}$, the λ_2-based algorithm does much better (i.e., it can solve the distinguishing problem for a smaller value of d).

6.3 Improvements for Dense vs Random

For the Dense vs Random question addressed in section 3.2, we consider a natural SDP relaxation for the densest k subgraph problem considered by [FS97], and show that this performs better than the algorithm presented earlier in the parameter range $k > \sqrt{n}$. The SDP is as follows

[3] which we assume, is the eigenvector corresponding to λ_1 – this is not quite true, since the graph is not of uniform degree, but it turns out this is not a critical issue.

$$\max \sum_{(i,j)\in E(G)} X_{ij} \qquad \text{subject to} \tag{12}$$

$$\sum_i X_{ii} = k \tag{13}$$

$$\sum_j X_{ij} = kX_{ii} \qquad \text{for all } i \tag{14}$$

$$X_{ij} \le X_{ii} \qquad \text{for all } i,j \tag{15}$$

$$X \succeq 0 \tag{16}$$

This is a relaxation for the problem and it is easy to see that there exists an SDP solution of value $|E(H)|$, where H is a k-subgraph. However we show that the SDP value for a random graph is upper bounded by exhibiting a suitable dual solution.

Theorem 6.1. *For a random graph $G(n,p)$ of average degree D ($D = np$), the value of the SDP is at most $k(\sqrt{D} + k^2 D/n)$ w.h.p.*

Proof. (of theorem 6.1) Let us consider the dual. We have the variables t (eq. 13), y_i (eq. 14), z_{ij} (eq. 15). The dual is

$$\begin{aligned} \min\ kt \qquad & \text{subject to} \\ U - A \succeq 0 \qquad & \\ U_{ii} = t - ky_i - \sum_j z_{ij} \qquad & \text{for all } i \\ U_{ij} = y_i + z_{ij} \qquad & \text{for all } i,j \end{aligned}$$

For a random graph, note that

$$\frac{D}{n} J - A + \lambda_2 I \succeq 0$$

where J is the all-ones matrix (It is easy to see this considering two cases: the all ones vector, and vectors orthogonal to it). Hence if we set $y_i = \frac{D}{n}$, $t = \lambda_2 + \frac{kD}{n}$ and the rest to 0, we get a feasible dual solution of value $\frac{k^2 D}{n} + k\lambda_2 \le k^2 \frac{D}{n} + k\sqrt{D}$.

In the last step, we used the fact that the second eigenvalue is at most $\sqrt{np}$ w.h.p. □

As an immediate corollary, we have

Corollary 6.2. *The SDP (12) can be used to distinguish between a $G(n,p)$ random graph with average degree $D = np$ and a graph with a k-subgraph of density $\sqrt{D} + kD/n$.*

Note that the distinguishing guarantee in Corollary 6.2 is better than the log-density guarantee from Section 3.2 when $k > \sqrt{n}$.

References

[ABBG10] Sanjeev Arora, Boaz Barak, Markus Brunnermeier, and Rong Ge. Computational complexity and information asymmetry in financial products. In *Proceedings of the First Symposium on Innovations in Computer Science (ICS)*, 2010.

[ABW08] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public key cryptography from different assumptions. Preliminary version as cryptology eprint report 2008/335 by Barak and Wigderson, 2008.

[AHI02] Yuichi Asahiro, Refael Hassin, and Kazuo Iwama. Complexity of finding dense subgraphs. *Discrete Appl. Math.*, 121(1-3):15–26, 2002.

[AKS98] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. pages 457–466, 1998.

[AS08] Noga Alon and Joel Spencer. *The Probabilistic Method*. Wiley-Interscience, third edition, 2008.

[BG09] Benjamin E. Birnbaum and Kenneth J. Goldman. An improved analysis for a greedy remote-clique algorithm using factor-revealing LPs. *Algorithmica*, 1:42–59, 2009.

[CHK09] Moses Charikar, MohammadTaghi Hajiaghayi, and Howard J. Karloff. Improved approximation algorithms for label cover problems. In *ESA*, volume 5757 of *Lecture Notes in Computer Science*, pages 23–34. Springer, 2009.

[Fei02] Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the 34th annual ACM Symposium on Theory of Computing (STOC'02)*, pages 534–543. ACM Press, 2002.

[FK81] Zoltán Füredi and János Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1:233–241, 1981.

[FKP01] Uriel Feige, Guy Kortsarz, and David Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[FL01] Uriel Feige and Michael Langberg. Approximation algorithms for maximization problems arising in graph partitioning. *J. Algorithms*, 41(2):174–211, 2001.

[FS97] Uriel Feige and Michael Seltser. On the densest k-subgraph problems. Technical Report CS97-16, Weizmann Institute of Science, Rehovot, Israel, 1997.

[GGT89] Giorgio Gallo, Michael D. Grigoriadis, and Robert Endre Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18(1):30–55, 1989.

[Kho04] Subhash Khot. Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique. In *Proceedings of the 44th Annual IEEE Symposium on the Foundations of Computer Science (FOCS'04)*, pages 136–145, 2004.

[LS91] László Lovász and Alexander Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991.

[SW98] Anand Srivastav and Katja Wolf. Finding dense subgraphs with semidefinite programming. In *Proceedings of the International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 181–191, 1998.

A Simplifications

A.1 Lossless Simplifications

We present here a few simplifying assumptions which can be made with at most a constant factor loss in the approximation guarantee. We begin with a greedy algorithm which allows us to bound the maximum degree in G, taken from [FKP01] (Lemmas 3.2 and 3.3).

Lemma A.1. *There exists a polynomial time algorithm which, given a graph G containing a k-subgraph with average degree d, outputs a k-subgraph H' and an $(n - k/2)$-subgraph G' s.t.*

1. *For some $D > 0$, G' has maximum degree $\leq D$ and H' has average degree $\max\{\Omega(Dk/n), 1\}$.*
2. *Either G' contains a k-subgraph with average degree $\Omega(d)$, or H' has average degree $\Omega(d)$.*

Proof. Let U be the set of $k/2$ vertices of highest degree in G (breaking ties arbitrarily), and let $D = \min_{u \in U} \deg(u)$. Let U' be the set of $k/2$ vertices of G of highest degree into U. Let H' be the graph induced on $U \cup U'$. Observe that H' has $\Omega(Dk^2/n)$ edges and average degree $\Omega(Dk/n)$ (it also has average degree at least 1, assuming G has at least $k/2$ edges).

Now let G' be the subgraph induced on $V \setminus U$ (note that by definition, G' has maximum degree $\leq D$). Let H be the densest k-subgraph in G, and let α be the fraction of edges of H that are incident with vertices of U. If $\alpha \leq 1/2$, then the $dk/2$ of the edges in H remain in G' (and thus G' still has a k-subgraph with average degree $\Omega(d)$. On the other hand, if $\alpha \geq 1/2$ then it is not hard to see that the H' must have average degree $\Omega(d)$. □

We now justify the weakened requirement, that the algorithm should return a subgraph of size *at most* k (rather than exactly k).

Lemma A.2. *Given an algorithm which, whenever G contains a k-subgraph of average degree $\Omega(d)$ returns a k'-subgraph of average degree $\Omega(d')$, for some (non specific) $k' < k$), we can also find a (exactly) k-subgraph with average degree $\Omega(d')$ for such G.*

Proof. Apply the algorithm repeatedly, each time removing from G the edges of the subgraph found. Continue until the union of all subgraphs found contains at least k vertices. The union of subgraphs each of average degree d' has average degree at least d'. Hence there is no loss in approximation ratio if we reach k vertices by taking unions of smaller graphs. Either we have not removed half the edges from the optimal solution (and then all the subgraphs found – and hence their union – have average degree $\Omega(d')$), or we have removed half the edges of the optimal solution, in which case our subgraph has at least $dk/2$ edges.

Note that this algorithm may slightly overshoot (giving a subgraph on up to $2k$ vertices), in which case we can greedily prune the lowest degree vertices to get back a k-subgraph with the same average degree (up to a constant factor). □

We now treat the assumption that G (and hence H) is bipartite (as in the rest of the paper, we define the approximation ratio of an algorithm A as the maximum value attained by $|\mathrm{OPT}(G)|/|A(G)|$):

Lemma A.3. *Given an $f(n)$-approximation for DkS on n-vertex bipartite graphs, we can approximate DkS on arbitrary graphs within a $\Omega(f(2n))$-factor.*

Proof. Take two copies of the vertices of G, and connect copies of vertices (in two different sides) which are connected in G. Thus the densest $2k$-subgraph in the new bipartite graph has at least the same average degree as the densest k-subgraph in G. Now take the subgraph found by the bipartite DkS approximation on the new graph, and collapse the two sides (this cannot reduce the degree of any vertex in the subgraph). Note that the subgraph found may be a constant factor larger than k, in which case, as before, we can greedily prune vertices and lose a constant factor in the average degree. □

Finally, we note that it suffices to consider minimum degree in H, rather than average degree.

Lemma A.4. *Approximation algorithms for DkS lose at most a constant factor by assuming the densest k-subgraph has minimum degree d as opposed to average degree.*

Proof. Let H be the densest k-subgraph in G. Iteratively remove from H every vertex whose remaining degree into H is less than $d/2$. Since less than $kd/2$ edges are removed by this procedure, a subgraph on $k' \leq k$ vertices and minimum degree at least $d/2$ must remain. As noted in Lemma A.2, the fact that now we will be searching for subgraphs with $k' \neq k$ vertices can affect the approximation guarantee by at most a constant factor. □

A.2 Bounding the product kD

It may simplify some of our manipulations if we can assume that $kD \leq n$. While we do not formally make this assumption anywhere, it is often instructive to consider this case for illustrative purposes. In fact, we can make this assumption, however at the cost of introducing randomness to the algorithm, and, more significantly, losing an $O(\sqrt{\log n})$ factor in the approximation guarantee.

Assume that contrary to our assumption, $D > n/k$. In this case, take a random subgraph G' of G by keeping every edge with probability $n/(kD)$. The new maximum degree D' now satisfies roughly $D' = Dn/(kD) = n/k$ as desired. A standard probabilistic argument (involving the combination of a Chernoff bound and a union bound) shows that for every vertex induced subgraph, if d denotes its average degree in G, then its average degree in G' is at most $O((1 + dD'/D)(1 + \sqrt{\log n/(1 + dD'/D)}))$.

Let ρ be the approximation ratio that we are aiming at. For H, the densest k-subgraph, we may assume that $d \geq \rho\sqrt{\log n}kD/n$, as otherwise the greedy algorithm in Lemma A.1 provides a $\rho\sqrt{\log n}$ factor approximation. (This is where our argument loses more than a constant factor in the approximation ratio). Hence the average degree of H in G' is at least $\rho\sqrt{\log n}$. An approximation algorithm with ratio ρ will then find in G' a subgraph of average degree at least $\sqrt{\log n}$. Such a subgraph must have average degree a factor of D/D' higher in G, giving an approximation ratio of ρ.

B Solving the LP

We will show that the linear program DkS-LP$_t(G, k, d)$ can be solved in time $n^{O(t)}$ by giving an equivalent formulation with explicit constraints which has total size (number of variables and constraints) $n^{O(t)}$. We start by defining a homogenized version of the basic LP. We say that a vector $(y_1, \ldots, y_n, h)$ is a solution to **DkS-hom**$_t(G, k, d)$ if it satisfies:

$$\begin{aligned}
\sum_{i \in V} y_i \leq kh, \qquad & \text{and} \\
h \geq 0 \qquad & \\
\exists \{y_{ij} \mid i,j \in V\} \qquad & \text{s.t.} \\
\forall i \in V \qquad & \sum_{j \in \Gamma(i)} y_{ij} \geq d y_i \\
\forall i,j \in V \qquad & y_{ij} = y_{ji} \\
\forall i,j \in V \qquad & 0 \leq y_{ij} \leq y_i \leq h \\
\text{if } t > 1, \forall i \in V \qquad & \{y_{i1}, \ldots, y_{in}, y_i\} \in \mathbf{DkS\text{-}hom}_{t-1}(G,k,d)
\end{aligned}$$

Note now that the condition $(y_1, \ldots, y_n) \in \text{DkS-LP}_t(G,k,d)$ is now equivalent to $(y_1, \ldots, y_n, 1) \in \text{DkS-hom}_t(G,k,d)$. Moreover, we can flatten out the recursive definition of DkS-hom$_t$ to give a fully explicit LP on $n^{t+O(1)}$ variables and $n^{t+O(1)}$ constraints, which can be solved in time $n^{O(t)}$ by standard techniques.