Discrete Applied Mathematics 184 (2015) 130–138

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Parameterized clique on inhomogeneous random graphs☆

Tobias Friedrich, Anton Krohmer*

Institute for Mathematics and Computer Science, Friedrich-Schiller-Universität Jena, D-07743, Germany

ARTICLE INFO

Article history:
Received 19 May 2014
Received in revised form 8 October 2014
Accepted 15 October 2014
Available online 7 November 2014

Keywords:
Clique
Scale-free network
Power law graph
Inhomogeneous random graph

ABSTRACT

Finding cliques in graphs is a classical problem which is in general NP-hard and parameterized intractable. In typical applications like social networks or biological networks, however, the considered graphs are scale-free, i.e., their degree sequence follows a power law. Their specific structure can be algorithmically exploited and makes it possible to solve clique much more efficiently. We prove that on inhomogeneous random graphs with n nodes and power law exponent β, cliques of size k can be found in time $\mathcal{O}(n)$ for $\beta \geq 3$ and in time $\mathcal{O}(ne^{k^4})$ for $2 < \beta < 3$.

1. Introduction

The proliferation of scale-free network models has recently been propelled by experimental findings: Many real-world graphs like social networks, electricity maps, biological networks, co-author graphs, sex graphs, etc. have been found to exhibit similar properties even though they stem from vastly different fields and sources [8,19,21]. They all have a power law degree distribution, meaning that the number of vertices with degree k is proportional to $k^{-\beta}$, where β is a constant intrinsic to the network.

Following this discovery, an abundance of different theoretical models for these networks has been proposed, among which the probably most well known are the Preferential Attachment [3] and the Inhomogeneous Random Graphs [23]. There has been a significant body of research devoted to finding more similarities between these networks (e.g. low diameter, large clustering); there has been little work, however, on how to exploit these properties for algorithmic problems. In fact, many such problems like k-Clique that are believed to be intractable were originally inspired by scale-free networks—even though at the time the term "clique" was coined, the notion of scale-free networks did not yet exist [17].

It is therefore natural to investigate these real-world inspired problems on power law graphs. In this paper, we present different algorithms for finding fully connected subgraphs (cliques) of size k in inhomogeneous random graphs. It has been well studied that on general inputs, k-Clique is NP-complete [15]. An efficient polynomial time algorithm is therefore unlikely to exist. In fact, the combinatorial explosion of this problem is in both parameters n and k: A runtime $\mathcal{O}(f(k)\cdot\text{poly}(n))$ that scales arbitrarily in the problem parameter k, but only polynomially in the input size n is considered to be unachievable, as this problem is also $W[1]$-complete [11].

Recent findings, however, suggest that things look differently on scale-free networks: Although there can be cliques of polynomial size [4], Janson, Łuczak, and Norros [14] proved formally that one can retrieve a $1 - o(1)$ approximation of the largest clique with high probability when given the underlying theoretical model of the graph. Eppstein and Strash [10] furthered this intuition experimentally by enumerating all maximal cliques on several data sets in feasible time. In this paper,

☆ This paper improves its conference version [13].

* Corresponding author. Tel.: +49 3641 946325; fax: +49 3641 946322.
E-mail address: anton.krohmer@uni-jena.de (A. Krohmer).

http://dx.doi.org/10.1016/j.dam.2014.10.018
0166-218X/

we show that there exist *exact* algorithms that only need the graph as input and no further metainformation. In particular – in contrast to previous theoretical results – these algorithms do not require access to the generally unknown node weights.

Results. The behavior of scale-free networks depends significantly on the exponent β of the power law degree distribution. In the case $\beta > 3$ (e.g. sexual contacts, citations or electronic circuits [8,20]), the expected maximal size of a clique is constant [5,14]. This implies that large cliques are unlikely, but does not imply a fast algorithm that always answers correctly. The difficulty is certifying a negative answer. We prove the following theorem.

Theorem 1. *The k-*Clique *problem can be solved in expected time $\mathcal{O}(n)$ on inhomogeneous random graphs with power law exponent $\beta \geq 3$.*

Throughout this paper, all asymptotic notation is given in both parameters n and k, but hides constants like average degree δ and power law exponent β. Consequently, the runtime of our algorithm does not depend on k and is therefore asymptotically optimal. Our algorithm is deterministic and always returns the correct answer. Moreover, it does not use any underlying information of the model (e.g. weights). Note that this theorem implies that k-Clique, which is NP-complete in general, in this setting becomes avgP, which is the average-case analog of P [16]. The best result so far was $\mathcal{O}(n^4)$ by Janson et al. [14] and an algorithm with $\mathcal{O}(n^2)$ runtime by the conference version [13] of this paper. Note that an application of a Markov bound yields the following high probability bound on the runtime.

Corollary 2. *Let $f(n)$ be any function such that grows asymptotically slower than n, i.e. $f(n) \in \omega(n)$. Then, the k-*Clique *problem can be solved in time $f(n)$ on inhomogeneous random graphs with power law $\beta \geq 3$ with high probability.*

On the other hand, many scale-free networks (e.g. co-actors, protein interactions, internet, peer-to-peer [20]) have a power law exponent β with $2 < \beta < 3$. In this case, the expected maximal size of a clique diverges [5,14] and there exists a *core*. The core is a subgraph that has a diameter of $\mathcal{O}(\log\log n)$ and contains a dense Erdős–Rényi graph [7,23]. As this is a known hard problem, we cannot expect similarly good results as for $\beta \geq 3$. We prove the following theorem.

Theorem 3. *The k-*Clique *problem can be solved in time $\mathcal{O}(n \exp(k^4))$ with overwhelming*[1] *probability on inhomogeneous random graphs with power law exponent $2 < \beta < 3$.*

While in general k-Clique is not believed to be parameterized tractable, i.e. in FPT, the above theorem shows that in this setting k-Clique is typically parameterized tractable, i.e. in typFPT, which is an average-case analog of FPT as defined in [12]. We are confident that this result extends to exponents $\beta \leq 2$, but as those networks exhibit significantly different properties and are rare in reality compared to the above mentioned cases, we did not investigate them formally.

2. Preliminaries

In order to achieve high general validity, we use the inhomogeneous random graph model of van der Hofstad [23], which generalizes the models of Chung–Lu [7,1,2] and Norros–Reittu [22] as well as the generalized random graphs. The model has two adjustable parameters: the exponent of the scale-free network β and the average degree δ. Depending on these two parameters, each node i has a weight w_i. This determines the edge probability $p_{ij} := \Pr[\{i,j\} \in E]$, which intuitively should be set proportional to $w_i w_j$.

Weights w_i. A simple way to fix the weights would be for example $w_i = \delta(n/i)^{\frac{1}{\beta-1}}$. However, we aim for a more general setting and proceed differently. Given the weights w_i, we can use the empirical complementary cumulative distribution function (CCDF) $F_n(w) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[w_i \geq w]$. This gives us $F_n(w) = \Pr[W \geq w]$, where W is a random variable chosen uniformly from the weights $w_1, \ldots, w_n$. Instead of fixing w_i, it is now easier to start from $F_n(w)$ and assume the following.

Definition 4 (*Power-Law Weights*)**.** We say that an empirical CCDF $F_n(w)$ follows the *power law with exponent β*, if there exist two positive constants α_1, α_2 such that

$$\alpha_1 w^{-\beta+1} \leq F_n(w) \leq \alpha_2 w^{-\beta+1}.$$

Then, we require the weights $w_1, \ldots, w_n$ to have the empirical CCDF $F_n(w)$. Following van der Hofstad [23], we moreover require that the empirical CCDF F_n satisfies the following properties.

Definition 5 (*Regularity Conditions for Node Weights*)**.**

(1) *Weak convergence of node weights.* There exists a function F such that $\lim_{n\to\infty} F_n(x) = F(x)$.
(2) *Convergence of average node weight.* Let W_n and W have distribution functions F_n and F, respectively. Then, it holds that $\lim_{n\to\infty} \mathbb{E}[W_n] = \mathbb{E}[W]$. Furthermore, $\mathbb{E}[W] > 0$.

[1] We use the terms *high probability* for probability $1 - o(1)$, *negligible probability* for probability $1/f(n)$, and *overwhelming probability* for probability $1 - 1/f(n)$, where $f(n)$ is any superpolynomially increasing function.

The regularity of F_n guarantees that the intuition $F_n(w) = \Pr[W \geq w]$ indeed holds. Furthermore, it guarantees that the average degree in the inhomogeneous random graphs converges, and that the largest weight is asymptotically bounded by $o(n)$, i.e. $\max_{i\in\{1,\dots,n\}} w_i = o(n)$. Both assumptions are sufficient to generate a scale-free network [23].

Edge probability p_{ij}. Other inhomogeneous random graph models use the following definitions:

$$p_{ij} = \min\left\{\frac{w_i w_j}{\sum_{k=1}^{n} w_k}, 1\right\} \qquad \text{(Chung–Lu)}$$

$$p_{ij} = \frac{w_i w_j}{\sum_{k=1}^{n} w_k + w_i w_j} \qquad \text{(Generalized Random Graph)}$$

$$p_{ij} = 1 - \exp\left\{-\frac{w_i w_j}{\sum_{k=1}^{n} w_k}\right\} \qquad \text{(Norros–Reittu).}$$

Since we restrict the weights w_i in Definition 4, it is possible to encompass these edge probability functions with the following definition.

Definition 6. We call p_{ij} the *edge probability* between nodes i and j of the inhomogeneous random graph, if it is 0 for $i = j$, and otherwise fulfills

$$p_{ij} = \mathcal{O}\left(\frac{w_i w_j}{n}\right) \quad \text{and} \quad p_{ij} = \Omega\left(\frac{w_i w_j}{n + w_i w_j}\right).$$

In order to see that for our weights w_i this is a generalization of all aforementioned scale-free random graph models, we observe that when $\beta \geq 2$, $w_{\min} = \Theta(1)$ and $w_{\min}^{-\beta+2} > w_{\max}^{-\beta+2}$ and compute

$$\sum_{k=1}^{n} w_k = n \cdot \mathbb{E}[W] = n \cdot \int_{w_{\min}}^{w_{\max}} F_n(w)\,\mathrm{d}w = \Theta(n \cdot w_{\min}^{-\beta+2}) = \Theta(n). \tag{1}$$

Notation. We use $\mathcal{G}_{\mathcal{SF}}(\beta)$ to refer to the probability space of inhomogeneous random graphs that were created as described above, and G to represent a graph drawn from $\mathcal{G}_{\mathcal{SF}}(\beta)$. By $\deg(v, G)$ we refer to the degree of a node v in a graph G. Whenever it is clear from the context which graph we are referring to, we write μ_i for the expected degree of node i. For the purpose of the analysis of the algorithms, the n vertices are identified by $1, \dots, n$ such that $w_i \leq w_j$ whenever $i \leq j$. We point out that this is only to simplify our presentation and our algorithms do not use this implicit ordering. Finally, we use the induced subgraph $G_i := G[i, \dots, n]$ that describes an inhomogeneous random graph $G \leftarrow \mathcal{G}_{\mathcal{SF}}$ where nodes $j \notin \{i, \dots, n\}$ have been subsequently removed from the node set.

Expected degree. It is useful to observe that the expected degree of each node can be asymptotically upper bounded by its weight:

$$\mathbb{E}[\deg(i, G)] = \sum_{j=1}^{n} p_{ij} = \mathcal{O}\Big(\frac{w_i}{n}\sum_{j=1}^{n} w_j\Big) = \mathcal{O}(w_i). \tag{2}$$

This bound is in fact tight, as shown by the following lemma.

Lemma 7. *Let $G = (V, E)$ be a random graph drawn from $\mathcal{G}_{\mathcal{SF}}(\beta)$. Then, $\mathbb{E}[\deg(i, G)] = \Theta(w_i)$ for $\beta \geq 2$.*

Proof. Recall that we write μ_i for the expected degree $\mathbb{E}[\deg(i, G)]$ of node i. It is left to prove $\mu_i = \Omega(w_i)$. To this end, we compute

$$\begin{aligned}\mu_i &= \sum_{j=1}^{n} p_{ij} = \Omega\left(\sum_{j=1}^{n} \frac{w_i w_j}{n + w_i w_j}\right) \\ &= \Omega\left(\sum_{\substack{j\in\{1,\dots,n\} \\ w_i w_j < f(n)}} \frac{w_i w_j}{n\left(1 + \frac{w_i w_j}{n}\right)}\right) = \Omega\left(\frac{w_i}{n}\sum_{\substack{j\in\{1,\dots,n\} \\ w_i w_j < f(n)}} w_j\right).\end{aligned}$$

In the third step above, we remove all summands that do not fulfill the requirement $w_i w_j < f(n)$. For $f(n)$, we choose a function which lies asymptotically between w_i and n, i.e., it fulfills both $f \in o(n)$ and $w_i \in o(f)$. Such a function indeed

exists: Since we assumed that $w_j = o(n)$ for all j, we can set $f(n) := \sqrt{w_i n}$. Then $w_i \in o(f)$, since $w_i/\sqrt{w_i n} = \sqrt{w_i/n} \to 0$. Conversely, $f \in o(n)$ holds by the same argument.

We compute the resulting sum by looking at the random variable W that is uniformly chosen from $w_1, \dots, w_n$:

$$\begin{aligned}
\mu_i &= \Omega\left(\frac{w_i}{n} \cdot \sum_{\substack{j\in\{1,\dots,n\}\\ w_i w_j < f(n)}} w_j\right) = \Omega\left(w_i \cdot \mathbb{E}\left[W \cdot \mathbb{1}\left[W < \frac{f(n)}{w_i}\right]\right]\right)\\
&= \Omega\left(w_i \cdot \Pr\left[W < \frac{f(n)}{w_i}\right] \cdot \mathbb{E}\left[W \,\middle|\, W < \frac{f(n)}{w_i}\right]\right)\\
&= \Omega\left(w_i \cdot \underbrace{\left(1 - F\left(\frac{f(n)}{w_i}\right)\right)}_{=1-o(1)} \cdot \int_{w_1}^{w_n} 1 - \Pr\left[W < w \,\middle|\, W < \frac{f(n)}{w_i}\right] \mathrm{d}w\right).
\end{aligned}$$

The term $F(f(n)/w_i) = \Theta((w_i/f(n))^{\beta-1})$ is in $o(1)$ since $w_i \in o(f)$. We split the integral in two parts, and make a case distinction:

(1) $w < \frac{f(n)}{w_i}$. The conditional probability simplifies to

$$\frac{\Pr[W < w]}{\Pr[W < f(n)/w_i]} = \frac{1 - F_n(w)}{1 - F_n(f(n)/w_i)}.$$

(2) $w \geq \frac{f(n)}{w_i}$. Then, conditional probability yields 1.

We therefore can further simplify the expected degree as follows. Recall that $1 - F(f(n)/w_i) = 1 - o(1) = \Theta(1)$ vanishes in a constant.

$$\begin{aligned}
\mu_i &= \Omega\left(w_i \cdot \left(\int_{w_1}^{f(n)/w_i} 1 - \frac{1 - F_n(w)}{1 - F_n(f(n)/w_i)}\,\mathrm{d}w + \int_{f(n)/w_i}^{w_n} 1 - 1\,\mathrm{d}w\right)\right)\\
&= \Omega\left(\frac{w_i}{1 - F_n(f(n)/w_i)} \cdot \left(\int_{w_1}^{f(n)/w_i} F_n(w) - F_n(f(n)/w_i)\,\mathrm{d}w\right)\right)\\
&= \Omega\left(w_i \cdot \left[c_1(-w^{-\beta+2}) - c_2 w \cdot \left(\frac{f(n)}{w_i}\right)^{-\beta+1}\right]_{w_1}^{f(n)/w_i}\right)\\
&= \Omega\left(w_i \cdot \left(c_1(w_1^{-\beta+2}) - c_2\left(\frac{f(n)}{w_i}\right)^{-\beta+2}\right)\right)\\
&= \Omega(w_i). \quad \square
\end{aligned}$$

3. Analysis for power-law exponent $\beta \geq 3$

In this section, we describe the algorithm to solve the clique problem in $\mathcal{O}(n)$ on average whenever the scale-free network exhibits an exponent of $\beta \geq 3$. Note that this is a runtime improvement by a factor of n compared to the conference version [13].

We exploit the scale-free structure by processing low-degree nodes first. To this end, we repeat the following simple steps: Determine the smallest degree node v from the graph. If there is a $(k-1)$-subset of neighbors of v that is a clique, return the resulting k-clique. Otherwise, remove v from the graph. This suggests that when the algorithm reaches the high-degree nodes, the graph is almost empty, which means that those nodes too are of low degree.

To prove that this algorithm runs in expected linear time, we analyze an intermediary approach that orders the nodes by their weights instead of their degrees. Lemma 7 then shows that the degree of a node is a suitable surrogate for its weight.

Implementation details. We use adjacency lists to store the edges; and an array D of length n to remember which nodes have been deleted from the graph. Whenever we want to delete a node i during the algorithm, we set $D[i] := 1$. When checking for potential cliques during the algorithms, we simply omit nodes i for which $D[i] = 1$.

A careful implementation allows for efficiently finding the node with minimum degree: An array A of length n stores in each position i all nodes of degree i. By memorizing an index in the array, one can extract the currently smallest degree node in constant time. To remove the currently smallest node, we have to update A at all positions of its neighbors; the overall runtime for updating the graph is proportional to the sum of all degrees. By Eq. (2), this is $\mathcal{O}(n)$.

Using this, we show that the greedy algorithm has an overall expected runtime of $\mathcal{O}(n)$. We begin by examining the expected degree of a node i when all nodes j with smaller weight are removed from the graph.

Lemma 8. *Let $G = (V, E)$ be a random graph drawn from $\mathcal{G}_{\mathcal{SF}}(\beta)$, where $\beta \geq 3$, and let $G_i = G[i, \dots, n]$. Then, we have*

$$\mathrm{E}[\deg(i, G_i)] = \mathcal{O}(1).$$

Proof. We use the indicator variable $\mathbb{1}[\{i,j\} \in E]$, which attains value 1 if $\{i,j\}$ is an edge and 0 otherwise. Whenever possible, we hide constants in $\mathcal{O}(1)$.

$$\mathrm{E}[\deg(i, G_i)] = \sum_{j=i+1}^{n} \mathrm{E}[\mathbb{1}[\{i,j\} \in E]] = \sum_{j=i+1}^{n} p_{ij} = \mathcal{O}\Big(\frac{w_i}{n} \sum_{j=1}^{n} w_j \cdot \mathbb{1}[w_j > w_i]\Big).$$

We now use the random variable W as described in Eq. (1), yielding

$$\begin{aligned}
\mathrm{E}[\deg(i, G_i)] &= \mathcal{O}\big(w_i \cdot \mathrm{E}[W \cdot \mathbb{1}[W \geq w_i]]\big) \\
&= \mathcal{O}\big(w_i \cdot \Pr[W \geq w_i] \cdot \mathrm{E}[W \mid W \geq w_i]\big) \\
&= \mathcal{O}\Big(w_i \cdot F_n(w_i) \cdot \int_{w_1}^{w_n} \Pr[W \geq w \mid W \geq w_i]\,\mathrm{d}w\Big).
\end{aligned}$$

To determine the probability that a weight W drawn uniformly at random admits $W \geq w$ given that $W \geq w_i$ holds, we distinguish two cases:

(1) $w \leq w_i$: Since we know that $W \geq w_i$, it is also larger than w.
(2) $w > w_i$: The conditional probability simplifies to $\frac{\Pr[W \geq w]}{\Pr[W \geq w_i]} = \frac{F_n(w)}{F_n(w_i)}$.

This simplifies the integral and yields

$$\begin{aligned}
\mathrm{E}[\deg(i, G_i)] &= \mathcal{O}\left(w_i \cdot F_n(w_i) \cdot \left(\int_{w_1}^{w_i} 1\,\mathrm{d}w + \int_{w_i}^{w_n} \frac{F_n(w)}{F_n(w_i)}\,\mathrm{d}w\right)\right) \\
&= \mathcal{O}\left(w_i(w_i - w_1) \cdot F_n(w_i) + \frac{w_i}{-\beta+2}\left[w^{-\beta+2}\right]_{w_i}^{w_n}\right) \\
&= \mathcal{O}\left(w_i^{-\beta+3} + w_i^{-\beta+3} - w_i w_n^{-\beta+2}\right) \\
&= \mathcal{O}(w_i^{-\beta+3}).
\end{aligned}$$

For $\beta \geq 3$, this term is constant, as the weights w_i are in $\Omega(1)$. □

Lemma 8 proves that a node i has constant degree if all nodes with smaller weight have been deleted from the graph. If the greedy algorithm ordered the nodes by increasing weights instead of smallest degree, this would be sufficient for proving linear runtime. As the weights of the model are not given, however, we show that processing the smallest degree node in a graph is as good as ordering the nodes by weight.

Proof of Theorem 1. Let $\tilde{G}_i$ be an induced subgraph of G such the greedy algorithm processes node i next. The node with smallest degree in $\tilde{G}_i$ is therefore node i.

Let T be the running time of the greedy algorithm. The runtime for processing a single node i during the algorithm is denoted by T_i. We recall that extracting the minimum degree node n times runs in overall $\mathcal{O}(n)$ time, which we omit from the computation below.

The number of $(k-1)$-subsets of vertices a node with x neighbors allows is $\binom{x}{k-1} \leq 2^x$, and the time needed to check whether a subset is a clique is $(k-1)^2 \leq x^2$. Thus, we get

$$\begin{aligned}
\mathbb{E}[T] &= \mathcal{O}\left(\sum_{i=1}^{n} \mathbb{E}[T_i]\right) \\
&\leq \mathcal{O}\left(\sum_{i=1}^{n} \sum_{x=1}^{n} 2^x \cdot x^2 \cdot \Pr\Big[\deg(i, \tilde{G}_i) \geq x\Big]\right). \qquad (3)
\end{aligned}$$

Now we show that $\Pr[\deg(i, \tilde{G}_i) \geq x]$ decreases exponentially. Let s be the smallest weight node in $\tilde{G}_i$. It is possible (but not necessary) that $i = s$. Consider now the graph G_s which is the induced subgraph of G where all nodes with weight less than w_s have been removed. Since s is also the smallest weight node in $\tilde{G}_i$, we have $\tilde{G}_i \subseteq G_s$. Using that i is the smallest degree

node in $\tilde{G}_i$, we can conclude

$$\Pr[\deg(i, \tilde{G}_i) \geq x] \leq \Pr[\deg(s, \tilde{G}_i) \geq x] \leq \Pr[\deg(s, G_s) \geq x]. \tag{4}$$

Recall that the expected degree $\mathbb{E}[\deg(s, G_s)]$ is denoted by μ_s. As $\deg(s, G_s)$ is a sum of independent binary variables, we can apply a Chernoff bound [9] and obtain

$$\Pr[\deg(s, G_s) \geq x] \leq \left(\exp\left(\frac{x}{\mu_s} - 1\right) \Big/ \left(\frac{x}{\mu_s}\right)^{x/\mu_s}\right)^{\mu_s}.$$

By Lemma 8, μ_s is constant for all s. This leaves us with

$$\mathbb{E}[T] = \mathcal{O}\left(\sum_{i=1}^{n} \sum_{x=1}^{\infty} (2e\mu_s)^x x^{2-x}\right),$$

where $s = s(i)$. Since the inner sum converges to a constant, $\mathbb{E}[T] = \mathcal{O}(n)$. □

Let us briefly discuss the implications of this result. We concluded that the expected runtime of the greedy algorithm is linear. This, however, does not imply that the runtime is also linear with high probability. In fact, Corollary 2 only states the slightly weaker result that *any* runtime $f(n)$ which is asymptotically slower than $\mathcal{O}(n)$ can be achieved with high probability. Because of the intricate dependencies between the subgraphs $\tilde{G}_i$ and between the degrees of nodes it is difficult to show a better result.

The main issue in proving a runtime of $\mathcal{O}(n)$ with high probability lies in proving an analog of Lemma 8. It is hard to show that a processed node has constant degree with high probability since e.g. Chernoff only gives an upper bound of $\Theta(\log n)$. It is even harder to show that the bound holds for each step in the algorithm, since the degrees of subsequently processed nodes heavily depend on one another. As these complex techniques would only result in a minor improvement over the simple Markov bound in Corollary 2, we decided to leave this as an open question.

The role of the maximum degree. Another interesting question is how the maximum degree $d_{\max}$ of the graph affects the performance of our algorithm. Typically, $d_{\max}$ can be inferred from Definition 4 as follows.

$$F_n(w_n) = \Theta(w_n^{-\beta+1}) = \frac{1}{n} \Leftrightarrow w_n = \Theta(n^{\frac{1}{1-\beta}}).$$

Since $\mathbb{E}[d_{\max}] = \Theta(w_n)$, this gives the immediate answer that whenever $d_{\max} = \mathcal{O}(\sqrt{n})$ the greedy algorithm yields polynomial time and when $d_{\max} = \omega(\sqrt{n})$, it uses exponential time.

Finally, we would like to point out that the above proofs can easily be adjusted to show that the degeneracy [6] of an inhomogeneous random graph with exponent $\beta \geq 3$ is constant in expectation.

4. Analysis for power-law exponent $\beta \in (2, 3)$

Using the greedy algorithm of the previous section in the case $2 < \beta < 3$ would imply a superpolynomial runtime of $\text{poly}(n)^k$ since the neighborhood sizes in the algorithm increase with n. This only yields the result that k-Clique is in expectation in the parameterized class XP [11]. We therefore present a different algorithm to prove a better result. Instead of exploiting that there will be few edges and therefore cliques, we inspect the core of the graph, a dense subgraph consisting only of high-weight nodes. We show that the probability of the core containing a k-clique is high. This approach, however, is only feasible for small values of k which shows that k-Clique is parameterized tractable for the parameter k with high probability.

Partitioning algorithm. To find a k-clique, the partitioning algorithm first removes all nodes with degree below $\sqrt{n/\log\log n}$. The obtained subgraph $G' = (V', E')$ is arbitrarily partitioned into components of size k. Each component is then individually checked to determine if it is a clique. If no clique is found, the algorithm searches exhaustively all k-subsets of V. It is easy to see that this algorithm is correct.

We now want to prove that when k is small, the exhaustive search is triggered only with negligible probability. For this, we first show that there are polynomially many nodes in V' and that their mutual edge probabilities are $\geq 1/\log n$. We then use this to prove that one of the partitions is likely to be a clique. As in the previous section, the analysis would be much easier if, when choosing V', the algorithm was allowed to choose the nodes according to their weight, but unfortunately it only has access to their degree in the given graph. We do know, however, that the weight and the expected degree of a node are equal up to a constant factor. Using that, we can prove the two following Lemmas.

Lemma 9 (*Partitioning Algorithm Keeps Polynomially Many Nodes*)**.** *Let $\beta \in (2, 3)$, and $G = (V, E)$ be a scale-free graph drawn from $\mathcal{G}_{\mathcal{SF}}(\beta)$. Then,*

$$\Pr\left[\exists i > n - n^{\frac{3-\beta}{2}} : \deg(i, G) < \sqrt{n/\log\log n}\right] \leq \exp\left(-\Theta(\sqrt{n})\right).$$

Proof. For the sake of readability, let $a := \frac{3-\beta}{2}$. Then, using the union and a Chernoff bound, we have

$$\Pr\left[\exists i > n - n^a : \deg(i, G) < \sqrt{\tfrac{n}{\log\log n}}\right] < \sum_{i>n-n^a} \exp\left(-\tfrac{\mu_i}{2} + \sqrt{\tfrac{n}{\log\log n}}\right).$$

We now compute a lower bound on μ_i. Since $\mu_i = \Theta(w_i)$ by Lemma 7, we can use a lower bound on the weight of node i (where $i \geq n - n^a$). Recall that the weights $w_1, \ldots, w_n$ are monotonically increasing. Hence, there should only be $n - i$ nodes with a weight larger than w_i.

$$\frac{1}{n}(n-i) = F_n(w_i) = \Theta(w_i^{-\beta+1})$$
$$\Rightarrow w_i = \Theta(1) \cdot \left(\frac{n}{n-i}\right)^{\frac{1}{\beta-1}} \geq \Theta\left(n^{\frac{2-3+\beta}{2(\beta-1)}}\right) = \Theta\left(\sqrt{n}\right).$$

We now apply the lower bound for μ_i from Lemma 7 and obtain

$$\Pr\left[\exists i > n - n^a : \deg(i, G) < \sqrt{\tfrac{n}{\log\log n}}\right] < n \cdot \exp\left(\Theta\left(-\sqrt{n} + \sqrt{\tfrac{n}{\log\log n}}\right)\right).$$

The negative term is asymptotically larger than the positive term, and the factor $n = e^{\log n}$ vanishes in $\exp(-\Theta(\sqrt{n}))$. We can therefore conclude that

$$\Pr\left[\exists i > n - n^a : \deg(i, G) < \sqrt{n/\log\log n}\right] < \exp\left(-\Theta(\sqrt{n})\right). \quad \square$$

In a similar fashion, we show that the remaining nodes in the core are of high weight.

Lemma 10 (*Partitioning Algorithm Keeps Only High-Weight Nodes*)**.** *Let $\beta \in (2, 3)$, and $G = (V, E)$ be a scale-free graph drawn from $\mathcal{G}_{SF}(\beta)$. Then,*

$$\Pr\left[\exists i \in V' : w_i < \sqrt{\alpha_1 n/\log n}\right] \leq \exp\left(-\Theta(n^{\frac{1}{3}})\right).$$

Proof. For the sake of readability, let $b := n^{\frac{3-\beta}{2}} (\log n)^{\frac{\beta-1}{2}}$. We first compute a lower bound for the weight of all nodes $j \geq n - b$. The empirical CCDF gives us

$$\frac{1}{n}(n-j) = F_n(w) \geq \alpha_1 w_j^{-\beta+1}$$
$$\Rightarrow w_j \geq \left(\frac{\alpha_2 n}{n-j}\right)^{\frac{1}{\beta-1}} \geq \left(\alpha_2 n^{1-\frac{3-\beta}{2}} (\log n)^{-\frac{\beta-1}{2}}\right)^{\frac{1}{\beta-1}} > \sqrt{\alpha_1 n/\log n}.$$

It now suffices to show that every node $i < n - b$ is not contained in V' with high probability, i.e. we show

$$\Pr\left[\exists i < n - b : \deg(i, G) \geq \sqrt{n/\log\log n}\right] \leq \exp\left(-\Theta(n^{\frac{1}{3}})\right).$$

We once again proceed using the union and a Chernoff bound.

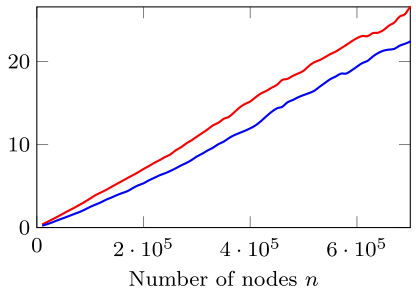
$$\Pr\left[\exists i < n - b : \deg(i, G) \geq \sqrt{n/\log\log n}\right] \leq \sum_{i<n-b} \exp\left(-\tfrac{\mu_i}{3} \min\left\{\left(\tfrac{\sqrt{n/\log\log n}}{\mu_i} - 1\right), \left(\tfrac{\sqrt{n/\log\log n}}{\mu_i} - 1\right)^2\right\}\right).$$

Lemma 7 gives $\mu_i = \Theta(1) \cdot w_i$. In a similar fashion as above, we can derive that for all $i < n-b$, $w_i < \alpha_2\sqrt{n/\log n}$. Therefore, we can simplify the term to

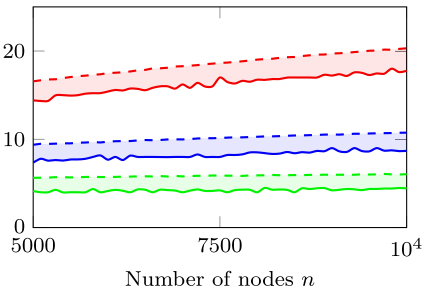
$$\begin{aligned}\Pr\left[\exists i < n - b : \deg(i, G) \geq \sqrt{n/\log\log n}\right] &\leq \sum_{i<n-b} \exp\left(-\Theta\left(\frac{\sqrt{n/\log\log n}}{\sqrt{n/\log n}} - 1\right)\sqrt{n/\log n}\right) \\ &= \exp\left(-\Theta(\sqrt{n/\log\log n})\right) < \exp(-\Theta(n^{\frac{1}{3}})). \quad \square\end{aligned}$$

It remains to show that the partitioning algorithm needs more than $k^2 n$ time only with negligible probability, if k is small enough. The idea is that the core V' of the scale-free network has larger edge probabilities than a dense Erdős–Rényi random graph, which is known to allow finding cliques fast [12].

Proof of Theorem 3. If $k > \log^{\frac{1}{3}} n$, then $n < e^{k^3}$, which implies that the exhaustive search of the partitioning algorithm runs in time $n^k < e^{k^4}$ and proves the claim. We can therefore assume $k \leq \log^{\frac{1}{3}} n$.



(a) Average runtime in seconds for the greedy algorithm sorting nodes by degrees (blue) and by weight (red) on scale-free graphs with $\beta = 3$.



(b) Average size of largest clique the partitioning algorithm finds in polynomial time (straight line) and average size of largest clique in the graph (dashed line) for $\beta = 2.3$, 2.5, and 2.7 (red, blue, and green).

Fig. 1. Experimental results for scale-free graphs with (a) $\beta = 3$ and (b) $2 < \beta < 3$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

As excluding unlikely events does not affect small failure probabilities, we can condition on the statements of Lemmas 9 and 10 and assume that there are more than $n^{\frac{3-\beta}{2}}$ nodes in V' and all nodes in V' have weight $\geq\sqrt{\alpha_1 n/\log n}$. This implies that $w_i w_j \geq \frac{\alpha_1 n}{\log n}$ and the edge probability between nodes $i, j \in V'$ is

$$p_{ij} = \Omega\left(\frac{w_i w_j}{n + w_i w_j}\right) = \Omega\left(\frac{1}{\log n}\right).$$

By choosing $g(n) = \frac{\log\log n - \Theta(1)}{\log n}$ suitably, we can bound the edge probability by $p_{ij} \geq n^{-g(n)} =: p$.

A k-partition is a k-clique with probability $\geq p^{\binom{k}{2}}$. The probability of not finding a clique before the exhaustive search is thus

$$\leq \left(1 - p^{\binom{k}{2}}\right)^{\left\lfloor n^{\frac{3-\beta}{2}}/k \right\rfloor} \leq \exp\left(-\left\lfloor \frac{n^{\frac{3-\beta}{2}}}{k} \right\rfloor p^{\binom{k}{2}}\right)$$

$$\leq \exp\left(-\frac{n^{\frac{3-\beta}{2}}}{2k} p^{\binom{k}{2}}\right) = \exp\left(-\frac{n^{\frac{3-\beta}{2} - g(n)\binom{k}{2}}}{2k}\right),$$

since we have $k \leq \log^{\frac{1}{3}} n \leq n^{\frac{3-\beta}{2}}/2$ for large n and therefore $\lfloor n^{\frac{3-\beta}{2}}/k \rfloor \geq n^{\frac{3-\beta}{2}}/k - 1 \geq n^{\frac{3-\beta}{2}}/(2k)$.

We now want to bound the expression $n^{\frac{3-\beta}{2} - g(n)\binom{k}{2}}$ from the above term. To this end, consider

$$g(n)\binom{k}{2} \leq g(n)k^2 = \frac{\log\log n - \Theta(1)}{\log n} k^2 = \mathcal{O}\left(\frac{(\log\log n)(\log n)^{2/3}}{\log n}\right) = o(1).$$

Therefore $n^{\frac{3-\beta}{2} - g(n)\binom{k}{2}} \geq n^{\frac{3-\beta}{2} - o(1)}$. It follows that the probability of doing the exhaustive search is

$$\leq \exp\left(-\frac{n^{\frac{3-\beta}{2} - g(n)\binom{k}{2}}}{2k}\right) \leq \exp\left(-\frac{n^{\frac{3-\beta}{2} - o(1)}}{2\log^{1/3} n}\right).$$

Since the inner expression grows polynomially, the algorithm enters exhaustive search with negligible probability, which proves the statement. □

5. Experiments

To check for the practical applicability of our theorems, we studied the empirical performance of the presented algorithms. To do so, we generated scale-free networks as described in Section 2 with Chung–Lu edge probabilities [18]. Our implementation is done in Java and run on a Dell PowerEdge M610 Blade Server with Intel Xeon E5620 CPUs (2.40 GHz). Each data point is averaged over 1000 runs.

Fig. 1(a) depicts the runtimes of the greedy algorithm versus n for $\beta = 3$ when ordering the nodes by smallest degree or by smallest weight, respectively. One can see that both these versions run in linear time; and the runtime is fast even for large graphs ($n = 600\,000$). Moreover, ordering the nodes by smallest degree slightly outperforms the approach that uses

(the practically not available) node weights. This is in line with Eq. (4) where we show that processing the smallest degree node is not worse than processing the smallest weight node. In fact, while the smallest weight node might be an outlier and have a degree of up to $\Theta(\log n)$, it is highly unlikely that the whole remaining graph contains no node with constant degree.

For the case $2 < \beta < 3$, the partitioning algorithm always finds the largest clique, but uses potentially exponential time in k. We are interested in the results that can be achieved in polynomial time (without the last step of an exhaustive search) as those are more relevant in practice. In Fig. 1(b) we compare, for different $\beta \in (2, 3)$, the average size of the largest clique found by the partitioning algorithm without exhaustive search against the average size of the largest clique. The plot demonstrates that the algorithm yields competitive results for different n and β; and even though the formal proof only guarantees finding a clique of size $k \leq \log^{1/3} n$ without exhaustive search, the empirical results suggest that we find a clique which is only off by a few nodes to the largest clique in polynomial time. We evaluated this algorithm only on small graphs since it becomes hard to extract the (optimal) maximum clique for larger n.

From a practical point of view, this algorithm can easily be improved heuristically. For instance, one can increase the size of a found clique by simply making it maximal, i.e. adding nodes that connect to all current nodes in the clique until no further addition is possible. Another way would be to group the partitions together by their degrees: Put all k highest degree nodes in one subset, the next k highest degrees in the next subset, and so forth. This should increase the chance for finding a k-clique in Chung–Lu graphs, although in our experience this intuition does not translate to real graphs. In fact, it might be more effective to just repeat the partitioning step with random partitions for a sufficient amount of time and take the best result.

6. Conclusion

Social networks are becoming ubiquitous. There is a significant body of research on the structural properties of such networks, but very little on how this can be exploited algorithmically. We have shown that for scale-free networks with n nodes and power law exponent β, the notoriously hard k-Clique problem admits a linear time algorithm for $\beta \geq 3$ (runtime $\mathcal{O}(n)$).

For the case $2 < \beta < 3$ – where cliques of polynomial size appear [14,5] – we proved that the problem admits an *exact* algorithm that runs in $\mathcal{O}(ne^{k^4})$ time with overwhelming probability. It remains an open question if an exact polynomial time algorithm can be found in this case. A more general open question is whether other NP-hard combinatorial problems (e.g. from bioinformatics) show similarly good average-case behavior on scale-free networks.

References

[1] W. Aiello, F. Chung, L. Lu, A random graph model for massive graphs, in: 32nd Symp. Theory of Computing, STOC, 2000, pp. 171–180.
[2] W. Aiello, F. Chung, L. Lu, A random graph model for power law graphs, Experiment. Math. 10 (1) (2001) 53–66.
[3] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[4] G. Bianconi, M. Marsili, Emergence of large cliques in random scale-free networks, Europhys. Lett. 74 (4) (2006) 740–746.
[5] G. Bianconi, M. Marsili, Number of cliques in random scale-free network ensembles, Physica D 224 (1–2) (2006) 1–6.
[6] A. Buchanan, J. Walteros, S. Butenko, P. Pardalos, Solving maximum clique in sparse graphs: an $o(nm + n2^{d/4})$ algorithm for d-degenerate graphs, Optim. Lett. (2013) 1–7.
[7] F. Chung, L. Lu, Connected components in random graphs with given expected degree sequences, Ann. Comb. 6 (2) (2002) 125–145.
[8] A. Clauset, C.R. Shalizi, M.E.J. Newman, Power-law distributions in empirical data, SIAM Rev. 51 (4) (2009) 661–703.
[9] D.P. Dubhashi, A. Panconesi, Concentration of Measure for the Analysis of Randomized Algorithms, Cambridge University Press, 2009.
[10] D. Eppstein, D. Strash, Listing all maximal cliques in large sparse real-world graphs, in: 10th Intl. Symp. Experimental Algorithms. SEA, in: LNCS, vol. 6630, Springer, 2011, pp. 364–375.
[11] M.R. Fellows, R.G. Downey, Parameterized Complexity, Springer, Verlag, 1998.
[12] N. Fountoulakis, T. Friedrich, D. Hermelin, On the average-case complexity of parameterized clique, Unpublished manuscript, 2012 http://arxiv.org/abs/1410.6400.
[13] T. Friedrich, A. Krohmer, Parameterized clique on scale-free networks, in: 23rd Intl. Symp. Algorithms and Computation, ISAAC, in: LNCS, vol. 7676, Springer, 2012, pp. 659–668.
[14] S. Janson, T. Łuczak, I. Norros, Large cliques in a power-law random graph, J. Appl. Probab. 47 (4) (2010) 1124–1135.
[15] R.M. Karp, Reducibility among combinatorial problems, in: Complexity of Computer Computations, Plenum Press, 1972, pp. 85–103.
[16] L.A. Levin, Average case complete problems, SIAM J. Comput. 15 (1) (1986) 285–286.
[17] R. Luce, A. Perry, A method of matrix analysis of group structure, Psychometrika 14 (2) (1949) 95–116.
[18] J. Miller, A. Hagberg, Efficient generation of networks with given expected degrees, in: 8th Intl. Workshop Algorithms & Models for the Web Graph, WAW, in: LNCS, vol. 6732, Springer, 2011, pp. 115–126.
[19] M. Mitzenmacher, A brief history of generative models for power law and lognormal distributions, Internet Math. 1 (2) (2004) 226–251.
[20] M.E.J. Newman, The structure and function of complex networks, SIAM Rev. 45 (2) (2003) 167–256.
[21] M.E.J. Newman, Power laws, Pareto distributions and Zipf's law, Contemp. Phys. 46 (5) (2005) 323–351.
[22] I. Norros, H. Reittu, On a conditionally Poissonian graph process, Adv. Appl. Probab. 38 (1) (2006) 59–75.
[23] R. van der Hofstad, Random graphs and complex networks, 2009. Available at www.win.tue.nl/~rhofstad/NotesRGCN.pdf.