Finding induced subgraphs in scale-free inhomogeneous random graphs

Ellen Cardinaels, Johan S.H. van Leeuwaarden, Clara Stegehuis

Eindhoven University of Technology

Abstract. We study the induced subgraph isomorphism problem on inhomogeneous random graphs with infinite variance power-law degrees. We provide a fast algorithm that determines for any connected graph H on k vertices if it exists as induced subgraph in a random graph with n vertices. By exploiting the scale-free graph structure, the algorithm runs in $O(nk)$ time for small values of k. We test our algorithm on several real-world data sets.

1 Introduction

The induced subgraph isomorphism problem asks whether a large graph G contains a connected graph H as an induced subgraph. When k is allowed to grow with the graph size n, this problem is NP-hard in general. For example, k-clique and k induced cycle, special cases of H, are known to be NP-hard [13, 20]. For fixed k, this problem can be solved in polynomial time $O(n^k)$ by searching for H on all possible combinations of k vertices. Several randomized and non-randomized algorithms exist to improve upon this trivial way of finding H [14, 25, 27, 29].

On real-world networks, many algorithms were observed to run much faster than predicted by the worst-case running time of algorithms. This may be ascribed to some of the properties that many real-world networks share [4], such as the power-law degree distribution found in many networks [1, 8, 19, 28]. One way of exploiting these power-law degree distributions is to design algorithms that work well on random graphs with power-law degree distributions. For example, finding the largest clique in a network is NP-complete for general networks [20]. However, in random graph models such as the Erdős-Rényi random graph and the inhomogeneous random graph, their specific structures can be exploited to design fixed parameter tractable (FPT) algorithms that efficiently find a clique of size k [10, 12] or the largest independent set [15].

In this paper, we study algorithms that are designed to perform well for the inhomogeneous random graph, a random graph model that can generate graphs with a power-law degree distribution [2,3,5,6,24,26]. The inhomogeneous random graph has a densely connected core containing many cliques, consisting of vertices with degrees $\sqrt{n \log(n)}$ and larger. In this densely connected core, the probability of an edge being present is close to one, so that it contains many complete graphs [18]. This observation was exploited in [11] to efficiently

determine whether a clique of size k occurs as a subgraph in an inhomogeneous random graph. When searching for *induced* subgraphs however, some edges are required not to be present. Therefore, searching for induced subgraphs in the entire core is not efficient. We show that a connected subgraph H can be found as an induced subgraph by scanning only vertices that are on the boundary of the core: vertices with degrees proportional to $\sqrt{n}$.

We present an algorithm that first selects the set of vertices with degrees proportional to $\sqrt{n}$, and then randomly searches for H as an induced subgraph on a subset of k of those vertices. The first algorithm we present does not depend on the specific structure of H. For general sparse graphs, the best known algorithms to solve subgraph isomorphism on 3 or 4 vertices run in $O(n^{1.41})$ or $O(n^{1.51})$ time with high probability [29]. For small values of k, our algorithm solves subgraph isomorphism on k nodes in linear time with high probability on inhomogeneous random graphs. However, the graph size needs to be very large for our algorithm to perform well. We therefore present a second algorithm that again selects the vertices with degrees proportional to $\sqrt{n}$, and then searches for induced subgraph H in a more efficient way. This algorithm has the same performance guarantee as our first algorithm, but performs much better in simulations.

We test our algorithm on large inhomogeneous random graphs, where it indeed efficiently finds induced subgraphs. We also test our algorithm on real-world network data with power-law degrees. There our algorithm does not perform well, probably due to the fact that the densely connected core of some real-world networks may not be the vertices of degrees at least proportional to $\sqrt{n}$. We then show that a slight modification of our algorithm that looks for induced subgraphs on vertices of degrees proportional to n^γ for some other value of γ performs better on real-world networks, where the value of γ depends on the specific network.

Notation. We say that a sequence of events $(\mathcal{E}_n)_{n\geq 1}$ happens with high probability (w.h.p.) if $\lim_{n\to\infty} \mathbb{P}(\mathcal{E}_n) = 1$. Furthermore, we write $f(n) = o(g(n))$ if $\lim_{n\to\infty} f(n)/g(n) = 0$, and $f(n) = O(g(n))$ if $|f(n)|/g(n)$ is uniformly bounded, where $(g(n))_{n\geq 1}$ is nonnegative. Similarly, if $\limsup_{n\to\infty} |f(n)|/g(n) > 0$, we say that $f(n) = \Omega(g(n))$ for nonnegative $(g(n))_{n\geq 1}$. We write $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ as well as $f(n) = \Omega(g(n))$.

1.1 Model

As a random graph null model, we use the inhomogeneous random graph or hidden variable model [2, 3, 5, 6, 24, 26]. Every vertex is equipped with a weight. We assume that the weights are i.i.d. samples from the power-law distribution

$$\mathbb{P}(w_i > k) = Ck^{1-\tau} \tag{1.1}$$

for some constant C and for $\tau \in (2, 3)$. Two vertices with weights w and w' are connected with probability

$$p(w, w') = \min\left(\frac{ww'}{\mu n}, 1\right), \tag{1.2}$$

where μ denotes the mean value of the power-law distribution (1.1). Choosing the connection probability in this way ensures that the expected degree of a vertex with weight w is w.

1.2 Algorithms

We now describe two randomized algorithms that determine whether a connected graph H is an induced subgraph in an inhomogeneous random graph and finds the location of such a subgraph if it exists. Algorithm 1 selects the vertices in the inhomogeneous random graph that are on the boundary of the core of the graph: vertices with degrees slightly below $\sqrt{\mu n}$. Then, the algorithm randomly divides these vertices into sets of k vertices. If one of these sets contains H as an induced subgraph, the algorithm terminates and returns the location of H. If this is not the case, then the algorithm fails. In the next section, we show that for k small enough, the probability that the algorithm fails is small. This means that H is present as an induced subgraph on vertices that are on the boundary of the core with high probability.

Algorithm 1 is similar to the algorithm in [12] designed to find cliques in random graphs. The major difference is that the algorithm to find cliques looks for cliques on all vertices with degrees larger than $\sqrt{f_1 \mu n}$ for some function f_1. This algorithm is not efficient for detecting other subgraphs than cliques, since vertices with high degrees will be connected with probability close to one.

Algorithm 1: Finding induced subgraph H (random search)

Input : H, $G = (V, E)$, μ, $f_1 = f_1(n)$, $f_2 = f_2(n)$.
Output: Location of H in G or fail.

```
Define n = |V|, I_n = [√(f1 μ n), √(f2 μ n)] and set V' = ∅.
for i ∈ V do
    if D_i ∈ I_n then V' = V' ∪ i;
end
Divide the vertices in V' randomly into ⌊|V'|/k⌋ sets S_1, ..., S_⌊|V'|/k⌋.
for j = 1, ..., ⌊|V'|/k⌋ do
    if H is an induced subgraph on S_j then return location of H;
end
```

The following theorem gives a bound for the performance of Algorithm 1 for small values of k.

Theorem 1. *Choose $f_1 = f_1(n) \geq 1/\log(n)$ and $f_1 < f_2 < 1$ and let $k < \log^{1/3}(n)$. Then, with high probability, Algorithm 1 detects induced subgraph H on k vertices in an inhomogeneous random graph with n vertices and weights distributed as in* (1.1) *in time $O(nk)$.*

Thus, for small values of k, Algorithm 1 finds an instance of H in linear time.

A problem with parameter k is called fixed parameter tractable (FPT) if it can be solved in $f(k)n^{O(1)}$ time for some function $f(k)$, and it is called typical FPT (typFPT) if it can be solved in $f(k)n^{g(n)}$ for some function $g(n) = O(1)$ with high probability [9]. As a corollary of Theorem 1 we obtain that the induced subgraph problem on the inhomogeneous random graph is in typFPT for any subgraph H, similarly to the k-clique problem on inhomogeneous random graphs [12].

Corollary 1. *The induced subgraph problem on the inhomogeneous random graph is in typFPT.*

In theory Algorithm 1 detects any motif on k vertices in linear time for small k. However, this only holds for large values of n, which can be understood as follows. In Lemma 2, we show that $|V'| = \Theta(n^{(3-\tau)/2})$, thus tending to infinity as n grows large. However, when $n = 10^7$ and $\tau = 2.5$, this means that the size of the set V' is only proportional to $10^{1.75} = 56$ vertices. Therefore, the number of sets S_j constructed in Algorithm 1 is also small. Even though the probability of finding motif H in any such set is proportional to a constant, this constant may be small, so that for finite n the algorithm almost always fails. Thus, for Algorithm 1 to work, n needs to be large enough so that $n^{(3-\tau)/2}$ is large as well.

The algorithm can be significantly improved by changing the search for H on vertices in set V'. In Algorithm 2 we propose a search for motif H similar to the Kashtan motif sampling algorithm [21]. Rather than sampling k vertices randomly, it samples one vertex randomly, and then randomly increases the set S by adding vertices in its neighborhood. This already guarantees the vertices in list S_j to be connected, making it more likely for them to form a specific connected motif together. In particular, we expand the list S_j in such a way that the vertices in S_j are guaranteed to form a spanning tree of H as a subgraph. This is ensured by choosing the list T^H that specifies at which vertex in S_j we expand S_j by adding a new vertex. For example, if $k = 4$ and we set $T^H = [1, 2, 3]$ we first add an edge to the first vertex, then we look for a random neighbor of the previously added vertex, and then we add a random neighbor of the third added vertex. Thus, setting $T^H = [1, 2, 3]$ ensures that the set S_j contains a path of length three, whereas setting $T^H = [1, 1, 1]$ ensures that the set S_j contains a star-shaped subgraph. Depending on which subgraph H we are looking for, we can define T^H in such a way that we ensure that the set S_j at least contains a spanning tree of motif H in Step 6 of the algorithm.

The selection on the degrees ensures that the degrees are sufficiently high so that probability of finding such a connected set on k vertices is high, as well as that the degrees are sufficiently low to ensure that we do not only find complete graphs because of the densely connected core of the inhomogeneous random graph. The probability that Algorithm 2 indeed finds the desired motif H in any check is of constant order of magnitude, similar to Algorithm 1. Therefore, the performance guarantee of both algorithms is similar. However, in practice Algorithm 2 performs much better, since for finite n, k connected vertices are more likely to form a motif than k randomly chosen vertices.

Algorithm 2: Finding induced subgraph H (neighborhood search)

Input : H, $G = (V, E)$, μ, $f_1 = f_1(n)$, $f_2 = f_2(n)$, s.
Output: Location of H in G or fail.

Define $n = |V|$, $I_n = [\sqrt{f_1 \mu n}, \sqrt{f_2 \mu n}]$ and set $V' = \emptyset$.
for $i \in V$ **do**
  if $D_i \in I_n$ **then** $V' = V' \cup i$;
end
Let G' be the induced subgraph of G on vertices V'.
Set T^H consistently with motif H .
for *j=1,. . . ,s* **do**
  Pick a random vertex $v \in V'$ and set $S_j = v$.
  while $|S_j| \neq k$ **do**
    Pick a random $v' \in N_{G'}(S_j[T^H[j]]) : v' \notin S_j$
    Add v' to S_j.
  end
  if *H is an induced subgraph on S_j* **then return** location of H;
end

The following theorem shows that indeed Algorithm 2 has similar performance guarantees as Algorithm 1.

Theorem 2. *Choose $f_1 = f_1(n) \geq 1/\log(n)$ and $f_1 < f_2 < 1$. Choose $s = \Omega(n^\alpha)$ for some $0 < \alpha < 1$, such that $s \leq n/k$. Then, Algorithm 2 detects induced subgraph H on $k < \log^{1/3}(n)$ vertices on an inhomogeneous random graph with n vertices and weights distributed as in* (1.1) *in time $O(nk)$ with high probability.*

The proofs of Theorem 1 and 2 rely on the fact that for small k, any subgraph on k vertices is present in G' with high probability. This means that after the degree selection step of Algorithms 1 and 2, for small k, any motif finding algorithm can be used to find motif H on the remaining graph G', such as the Grochow-Kellis algorithm [14], the MAvisto algorithm [27] or the MODA algorithm [25]. In the proofs of Theorem 1 and 2, we show that G' has $\Theta(n^{(3-\tau)/2})$ vertices with high probability. Thus, the degree selection step reduces the problem of finding a motif H on n vertices to finding a motif on a graph with $\Theta(n^{(3-\tau)/2})$ vertices, significantly reducing the running time of the algorithms.

2 Proof of Theorems 1 and 2

We prove Theorem 1 using two lemmas. The first lemma relates the degrees of the vertices to their weights. The connection probabilities in the inhomogeneous random graph depend on the weights of the vertices. In Algorithm 1, we select vertices based on their degrees instead of their unknown weights. The following lemma shows that the weights of the vertices in V' are close to their degrees.

Lemma 1. *Degrees and weights. Fix $\varepsilon > 0$, and define $J_n = [(1-\varepsilon)\sqrt{f_1\mu n}, (1+\varepsilon)\sqrt{f_2\mu n}]$. Then, for some $K > 0$,*

$$\mathbb{P}\left(\exists i \in V' : w_i \notin J_n\right) \leq Kn \exp\left(-\frac{\varepsilon^2(1-\varepsilon)}{2(1+\varepsilon)}\sqrt{f_1\mu n}\right). \tag{2.1}$$

Proof. Fix a vertex $i \in V$. Conditionally on the weight w_i of vertex i, $D_i \sim \mathrm{Poi}(w_i)$ [5, 16]. Then,

$$\begin{aligned}\mathbb{P}\left(w_i < (1-\varepsilon)\sqrt{f_1\mu n},\ D_i \in I_n\right) &= \frac{\mathbb{P}\left(D_i \in I_n \mid w_i < (1-\varepsilon)\sqrt{f_1\mu n}\right)}{\mathbb{P}\left(w_i < (1-\varepsilon)\sqrt{f_1\mu n}\right)} \\ &\leq \frac{\mathbb{P}\left(D_i > \sqrt{f_1\mu n} \mid w_i = (1-\varepsilon)\sqrt{f_1\mu n}\right)}{1 - C((1-\varepsilon)\sqrt{f_1\mu n})^{1-\tau}} \\ &\leq K_1 \mathbb{P}\left(D_i > \sqrt{f_1\mu n} \mid w_i = (1-\varepsilon)\sqrt{f_1\mu n}\right),\end{aligned} \tag{2.2}$$

for some $K_1 > 0$. Here the first inequality follows because for Poisson random variables $\mathbb{P}\left(\mathrm{Poi}(\lambda_1) > k\right) \leq \mathbb{P}\left(\mathrm{Poi}(\lambda_2) > k\right)$ for $\lambda_1 < \lambda_2$. We use that by the Chernoff bound for Poisson random variables

$$\mathbb{P}\left(X > \lambda(1+\delta)\right) \leq \exp\left(-h(\delta)\delta^2\lambda/2\right), \tag{2.3}$$

where $h(\delta) = 2((1+\delta)\ln(1+\delta) - \delta)/\delta^2$. Therefore, using that $h(\delta) \geq 1/(1+\delta)$ for $\delta \geq 0$ results in

$$\mathbb{P}\left(D_i > \sqrt{f_1\mu n} \mid w_i = (1-\varepsilon)\sqrt{f_1\mu n}\right) \leq \exp\left(-\frac{\varepsilon^2(1-\varepsilon)}{2(1+\varepsilon)}\sqrt{f_1\mu n}\right). \tag{2.4}$$

Combining this with (2.2) and taking the union bound over all vertices then results in

$$\mathbb{P}\left(\exists i : D_i \in I_n, w_i < (1-\varepsilon)\sqrt{f_1\mu n}\right) \leq K_1 n \exp\left(-\frac{\varepsilon^2(1-\varepsilon)}{2(1+\varepsilon)}\sqrt{f_1\mu n}\right). \tag{2.5}$$

The bound for $w_i > (1+\varepsilon)\sqrt{f_2\mu n}$ follows similarly. Combining this with the fact that $f_1 < f_2$ then proves the lemma. □

The second lemma shows that after deleting all vertices with degrees outside of I_n defined in Step 1 of Algorithm 1, still polynomially many vertices remain with high probability.

Lemma 2. *Polynomially many nodes remain. There exists $\gamma > 0$ such that*

$$\mathbb{P}\left(|V'| < \gamma n^{(3-\tau)/2}\right) \leq 2\exp\left(-\Theta(n^{(3-\tau)/2})\right). \tag{2.6}$$

Proof. Let $\mathcal{E}$ denote the event that all vertices $i \in V'$ satisfy $w_i \in J_n$ for some $\varepsilon > 0$, with J_n as in Lemma 1. Let W' be the set of vertices with weights in J_n. Under the event $\mathcal{E}$, $|V'| \leq |W'|$. Then, by Lemma 1

$$\mathbb{P}\left(|V'| < \gamma n^{(3-\tau)/2}\right) \leq \mathbb{P}\left(|W'| < \gamma n^{(3-\tau)/2}\right) + Kn \exp\left(-\frac{\varepsilon^2(1-\varepsilon)}{2(1+\varepsilon)}\sqrt{f_1\mu n}\right). \tag{2.7}$$

Furthermore,

$$\mathbb{P}\left(w_i \in J_n\right) = C((1-\varepsilon)\sqrt{f_1\mu n})^{1-\tau} - C((1+\varepsilon)\sqrt{f_2\mu n})^{1-\tau} \geq c_1(\sqrt{\mu n})^{1-\tau} \tag{2.8}$$

for some constant $c_1 > 0$ because $f_1 < f_2$. Thus, each of the n vertices is in set W' independently with probability at least $c_1(\sqrt{\mu n})^{1-\tau}$. Choose $0 < \gamma < c_1$. Applying the multiplicative Chernoff bound then shows that

$$\mathbb{P}\left(|W'| < \gamma n^{(3-\tau)/2}\right) \leq \exp\left(-\frac{(c_1-\gamma)^2}{2c_1} n^{(3-\tau)/2}\right), \tag{2.9}$$

which proves the lemma together with (2.7) and the fact that $\sqrt{f_1\mu n} = \Omega(n^{(3-\tau)/2})$ for $\tau \in (2,3)$. □

We now use these lemmas to prove Theorem 1.

Proof of Theorem 1. We condition on the event that V' is of polynomial size (Lemma 2) and that the weights are within the constructed lower and upper bounds (Lemma 1), since both events occur with high probability. This bounds the edge probability between any pair of nodes i and j in V' as

$$p_{ij} < \min\left(\frac{(1+\varepsilon)\sqrt{f_2\mu n}(1+\varepsilon)\sqrt{f_2\mu n}}{\mu n}, 1\right) = f_2(1+\varepsilon)^2, \tag{2.10}$$

so that $p_{ij} \leq p_+ = c_1 < 1$ if we choose ε small enough. Similarly,

$$p_{ij} > \min\left(\frac{(1-\varepsilon)^2\sqrt{f_1\mu n}^2}{\mu n}\right) = \Theta\left(\frac{1}{\log(n)}\right), \tag{2.11}$$

by our choice of f_1, so that $p_{ij} \geq p_- = c_2/\log(n)$. Let $E := |E_H|$ be the number of edges in H. We upper bound the probability of not finding H in one of the partitions of size k of V' as $1 - p_-^E(1-p_+)^{\binom{k}{2}-E}$. Since all partitions are disjoint we can upper bound the probability of not finding H in any of the partitions as

$$\mathbb{P}\left(H \text{ not in the partitions}\right) \leq \left(1 - p_-^E(1-p_+)^{\binom{k}{2}-E}\right)^{\left\lceil \frac{|V'|}{k} \right\rceil}. \tag{2.12}$$

Using that $E \leq k^2$, $\binom{k}{2} - E \leq k^2$ and that $1 - x \leq \mathrm{e}^{-x}$ results in

$$\mathbb{P}\left(H \text{ not in the partitions}\right) \leq \exp\left(-p_-^{k^2}(1-p_+)^{k^2}\left\lceil \frac{|V'|}{k} \right\rceil\right). \tag{2.13}$$

Since $|V'| = \Theta\left(n^{\frac{3-\tau}{2}}\right)$, $\lceil |V'|/k \rceil \geq dn^{\frac{3-\tau}{2}}/k$ for some constant $d > 0$. We fill in the expressions for p_- and p_+, with $c_3 > 0$ a constant

$$\mathbb{P}\left(H \text{ not in the partitions}\right) \leq \exp\left(-\frac{dn^{\frac{3-\tau}{2}}}{k}\left(\frac{c_3}{\log n}\right)^{k^2}\right). \tag{2.14}$$

Now apply that $k \leq \log^{\frac{1}{3}}(n)$. Then

$$\begin{aligned}\mathbb{P}\left(H \text{ not in the partitions}\right) &\leq \exp\left(-\frac{dn^{\frac{3-\tau}{2}}}{\log^{\frac{1}{3}} n}\left(\frac{c_3}{\log n}\right)^{\log^{\frac{2}{3}} n}\right) \\ &\leq \exp\left(-dn^{\frac{3-\tau}{2}-o(1)}\right).\end{aligned} \tag{2.15}$$

Hence, the inner expression grows polynomially such that the probability of not finding H in one of the partitions is negligibly small. The running time of the partial search is given by

$$\frac{|V'|}{k}\binom{k}{2} \leq \frac{n}{k}\binom{k}{2} \leq nk \leq n\mathrm{e}^{k^4}, \tag{2.16}$$

which concludes the proof for $k \leq \log^{1/3}(n)$. □

Proof of Corollary 1. If $k > \log^{\frac{1}{3}}(n)$, we can determine whether H is an induced subgraph by exhaustive search in time

$$\binom{n}{k}\binom{k}{2} \leq \frac{n^k}{k}\frac{k(k-1)}{2} \leq kn^k \leq k\mathrm{e}^{k^4} \leq n\mathrm{e}^{k^4}, \tag{2.17}$$

since for all sets of k vertices the presence or absence of $\binom{k}{2}$ edges needs to be checked. For $k \leq \log^{\frac{1}{3}}(n)$, Theorem 1 shows that the induced subgraph isomorphism problem can be solved in time $nk \leq n\mathrm{e}^{k^4}$. Thus, with high probability the induced subgraph isomorphism problem can be solved in $n\mathrm{e}^{k^4}$ time, which proves that it is in typFPT. □

Proof of Theorem 2. The proof of Theorem 2 is very similar to the proof of Theorem 1. The only way Algorithm 2 differs from Algorithm 1 is in the selection of the sets S_j. As in the previous theorem, we condition on the event that $|V'| = \Theta(n^{(3-\tau)/2})$ (Lemma 2) and that the weights of the vertices in G' are bounded as in Lemma 1.

The graph G' constructed in Step 5 of Algorithm 2 then consists of $\Theta(n^{(3-\tau)/2})$ vertices. Furthermore, by the bound (2.11) on the connection probabilities of all vertices in G', the expected degree of a vertex i in G' satisfies $\mathbb{E}\left[D_{i,G'}\right] = \Omega(n^{(3-\tau)/2}/\log(n))$. We can use similar arguments as in Lemma 1 to show that $D_{i,G'} = \Omega(n^{(3-\tau)/2}/\log(n))$ with high probability for all vertices in G'. Since G' consists of $\Theta(n^{(3-\tau)/2})$ vertices, $D_{i,G'} = O(n^{(3-\tau)/2})$ as well. This means that for $k < \log^{\frac{1}{3}}(n)$, Steps 8-11 are able to find a connected subgraph on k vertices with high probability.

We now compute the probability that S_j is disjoint with the previous $j-1$ constructed sets. The probability that the first vertex does not overlap with the previous sets is given by $1 - jk/|V'|$, since that vertex is chosen uniformly at random. The second vertex is chosen in a size-biased manner, since it is chosen by following a random edge. The probability that vertex i is added can therefore be bounded as

$$\mathbb{P}\left(\text{vertex } i \text{ is added}\right) = \frac{D_{i,G'}}{\sum_{s=1}^{|V'|} D_{s,G'}} \leq \frac{M\log(n)}{|V'|} \tag{2.18}$$

for some constant $M > 0$ by the conditions on the degrees. Therefore, the probability that S_j does not overlap with one of the previously chose jk vertices can be bounded from below by

$$\mathbb{P}\left(S_j \text{ does not overlap with previous sets}\right) \geq \left(1 - \frac{kj}{|V'|}\right)\left(1 - \frac{Mkj\log(n)}{|V'|}\right)^{k-1}. \tag{2.19}$$

Thus, the probability that all j sets do not overlap can be bounded as

$$\mathbb{P}\left(S_j \cap S_{j-1} \cdots \cap S_1 = \emptyset\right) \geq \left(1 - \frac{Mkj\log(n)}{|V'|}\right)^{jk}, \tag{2.20}$$

which tends to one when $jk = o(n^{(3-\tau)/4})$. Let s_{dis} denote the number of disjoint sets out of the s sets constructed in Algorithm 2. Then, when $s = \Omega(n^\alpha)$ for some $\alpha > 0$, $s_{\text{dis}} > n^\beta$ for some $\beta > 0$ with high probability, because $k < \log^{1/3}(n)$.

The probability that H is present as an induced subgraph is bounded similarly as in Theorem 1. We already know that $k-1$ edges are present. For all other $E-(k-1)$ edges of H, and all $\binom{k}{2} - E$ edges that are not present in H, we can again use (2.10) and (2.11) to bound on the probability of edges being present or not being present between vertices in V'. Therefore, we can bound the probability that H is not found similarly to (2.13) as

$$\begin{aligned}\mathbb{P}\left(H \text{ not in the partitions}\right) &\leq \mathbb{P}\left(H \text{ not in the disjoint partitions}\right)\\ &\leq \exp\left(-p_-^{k^2}\left(1-p_+\right)^{k^2} s_{\text{dis}}\right).\end{aligned}$$

Because $s_{\text{dis}} > n^\beta$ for some $\beta > 0$, this term tends to zero exponentially. The running time of the partial search can be bounded similarly to (2.16) as

$$s\binom{k}{2} \leq sk^2 = O(nk), \tag{2.21}$$

where we used that $s \leq n/k$. □

3 Experimental results

Figure 1 shows the fraction of times Algorithm 1 succeeds to find a cycle of size k in an inhomogeneous random graph on 10^7 vertices. Even though for large n Algorithm 1 should find an instance of a cycle of size k in step 7 of the algorithm with high probability, we see that Algorithm 1 never succeeds in finding one. This is because of the finite size effects discussed before.

Figure 2a also plots the fraction of times Algorithm 2 succeeds to find a cycle. We set the parameter $s = 10000$ so that the algorithm fails if the algorithm does not succeed to detect motif H after executing step 13 of Algorithm 2 10000 times. Because s gives the number of attempts to find H, increasing s may increase the success probability of Algorithm 2 at the cost of a higher running time. However,

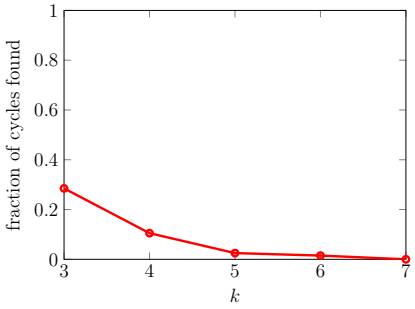


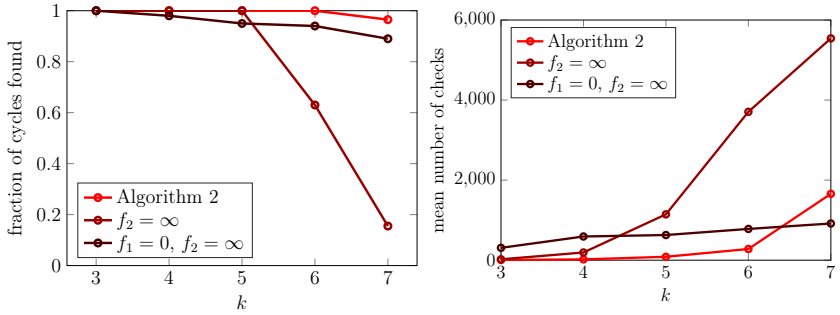
Fig. 1: The fraction of times step 7 in Algorithm 1 succeeds to find a cycle of length k on an inhomogeneous random graph with $N = 10^7$, averaged over 500 network samples with $f_1 = 1/\log(n)$ and $f_2 = 0.9$.

in Figure 2b we see that for small values of k, the mean number of times Step 13 is executed when the algorithm succeeds is much lower than 10000, so that increasing s in this experiment probably only has a small effect on the success probability. We see that Algorithm 2 outperforms Algorithm 1. Figure 2b also shows that the number of attempts needed to detect a cycle of length k is small for $k \leq 6$. For larger values of k the number of attempts increases. This can again be ascribed to the finite size effects that cause the set V' to be small, so that large motifs may not be present on vertices in set V'. We also plot the success probability when using different values of the functions f_1 and f_2. When only the lower bound f_1 on the vertex degrees is used, as in [11], the success probability of the algorithm decreases. This is because the set V' now contains many high degree vertices that are much more likely to form clique motifs than cycles or other connected motifs on k vertices. This makes $f_2 = \infty$ a very efficient bound for detecting clique motifs [11]. For the cycle motif however, we see in Figure 2b that more checks are needed before a cycle is detected, and in some cases the cycle is not detected at all.

Setting $f_1 = 0$ and $f_2 = \infty$ is also less efficient, as Figure 2a shows. In this situation, the number of attempts needed to find a cycle of length k is larger than for Algorithm 2 for $k \leq 6$.

3.1 Real network data

We now check Algorithm 2 on four real-world networks with power-law degrees: a Wikipedia communication network [22], the Gowalla social network [22], the Baidu online encyclopedia [23] and the Internet on the autonomous systems level [22]. Table 1 presents several statistics of these scale-free data sets. Figure 3 shows the fraction of runs where Algorithm 2 finds a cycle as an induced subgraph. We see that for the Wikipedia social network in Figure 3a, Algorithm 2 is more efficient than looking for cycles among all vertices in the network. For the Baidu online encyclopedia in Figure 3c however, we see that Algorithm 2



(a) Success probability

(b) Mean number of checks

Fig. 2: Results of Algorithm 2 on an inhomogeneous random graph with $N = 10^7$ for detecting cycles of length k. The parameters are chosen as $s = 10000$, $f_1 = 1/\log(n)$, $f_2 = 0.9$. The values are averaged over 500 generated networks.

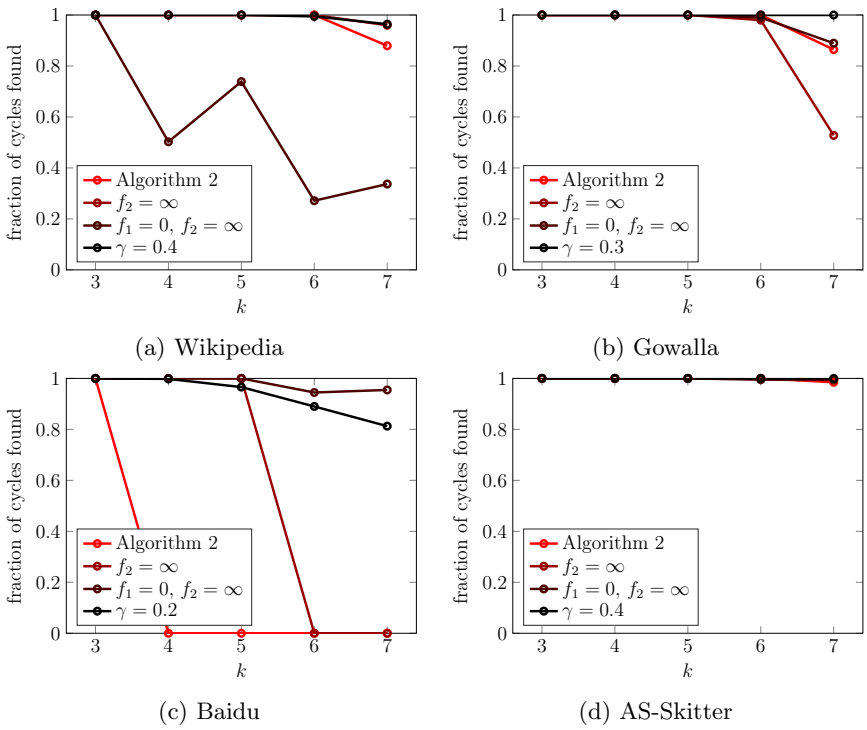
performs much worse than looking for cycles among all possible vertices. In the other two network data sets in Figures 3b and 3d the performance on the reduced vertex set and the original vertex set is almost the same. Figure 4 shows that in general, Algorithm 2 indeed seems to finish in fewer steps than when using the full vertex set. However, as Figure 4c shows, for larger values of k the algorithm fails almost always.

	n	E	τ
Wikipedia	2,394,385	5,021,410	2.46
Gowalla	196,591	950,327	2.65
Baidu	2,141,300	17,794,839	2.29
AS-Skitter	1,696,415	11,095,298	2.35

Table 1: Statistics of the data sets: the number of vertices n, the number of edges E, and the power-law exponent τ fitted by the method of [7].

These results show that while Algorithm 2 is efficient on inhomogeneous random graphs, it may not always be efficient on real-world data sets. This is not surprising, because there is no reason why the vertices of degrees proportional to $\sqrt{n}$ should behave like an Erdős-Rényi random graph, like in the inhomogeneous random graph. We therefore investigate whether selecting vertices with degrees in $I_n = [(\mu n)^\gamma / \log(n), (\mu n)^\gamma]$ for some other value of γ in Algorithm 2 leads to a better performance. Figure 3 and 4 show for every data set one particular value of γ that works well. For the Gowalla, Wikipedia and Autonomous systems network, this leads to a faster algorithm to detect cycles. Only for the Baidu network other values of γ do not improve upon randomly selecting from all vertices.

This indicates that for most networks, cycles do appear mostly on degrees with specific orders of magnitude, making it possible to sample these cycles faster. Unfortunately, these orders of magnitude may be different for different networks. Across all four networks, the best value of γ seems to be smaller than the value of 0.5 that is optimal for the inhomogeneous random graph.



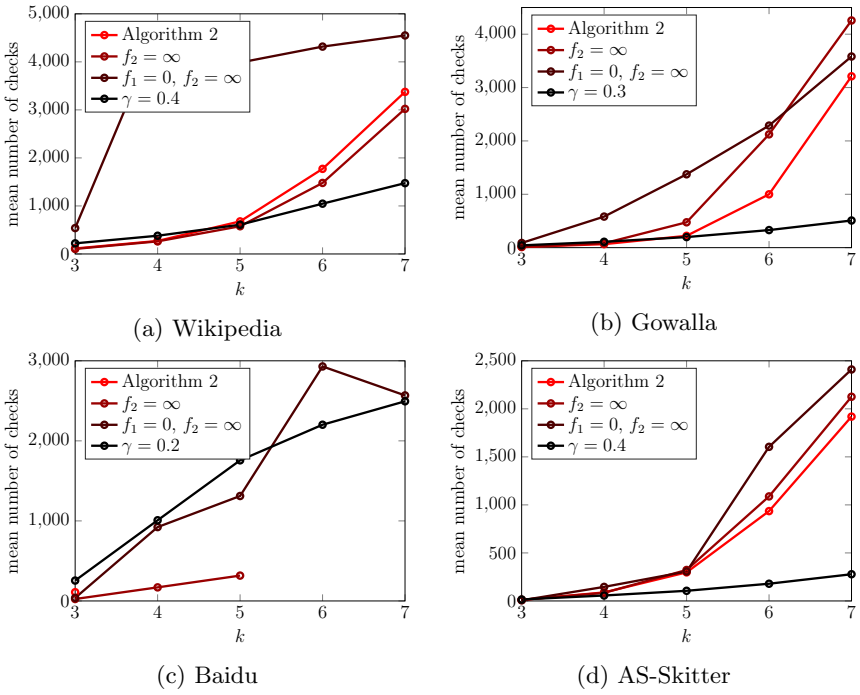
(a) Wikipedia (b) Gowalla

(c) Baidu (d) AS-Skitter

Fig. 3: The fraction of times Algorithm 2 succeeds to find a cycle on four large network data sets for detecting cycles of length k. The parameters are chosen as $s = 10000$, $f_1 = 1/\log(n)$, $f_2 = 0.9$. The black line uses Algorithm 2 on vertices of degrees in $I_n = [(\mu n)^\gamma/\log(n), (\mu n)^\gamma]$. The values are averaged over 500 runs of Algorithm 2.

4 Conclusion

We presented an algorithm which solves the induced subgraph problem on inhomogeneous random graphs with infinite variance power-law degrees in time $O(ne^{k^4})$ with high probability as n grows large. This algorithm is based on the observation that for fixed k, any subgraph is present on k vertices with degrees slightly smaller than $\sqrt{\mu n}$ with positive probability. Therefore, the algorithm



(a) Wikipedia (b) Gowalla

(c) Baidu (d) AS-Skitter

Fig. 4: The number of times step 12 of Algorithm 2 is invoked when the algorithm does not fail on four large network data sets for detecting cycles of length k. The parameters are chosen as $s = 10000$, $f_1 = 1/\log(n)$, $f_2 = 0.9$. The black line uses Algorithm 2 on vertices of degrees in $I_n = [(\mu n)^\gamma/\log(n), (\mu n)^\gamma]$. The values are averaged over 500 runs of Algorithm 2.

first selects vertices with those degrees, and then uses a random search method to look for the induced subgraph on those vertices.

We show that this algorithm performs well on simulations of inhomogeneous random graphs. Its performance on real-world data sets varies for different data sets. This indicates that the degrees that contain the most induced subgraphs of size k in real-world networks may not be close to $\sqrt{n}$. We then show that on these data sets, it may be more efficient to find induced subgraphs on degrees proportional to n^γ for some other value of γ. The value of γ may be different for different networks.

Our algorithm exploits that induced subgraphs are likely formed among $\sqrt{\mu n}$-degree vertices. However, certain subgraphs may occur more frequently on vertices of other degrees [17]. For example, star-shaped subgraphs on k vertices appear more often on one vertex with degree much higher than $\sqrt{\mu n}$ corresponding to the middle vertex of the star, and $k-1$ lower-degree vertices corresponding to the leafs of the star [17]. An interesting open question is whether there exist

better degree-selection steps for specific subgraphs than the one used in Algorithms 1 and 2.

Acknowledgements. The work of JvL and CS was supported by NWO TOP grant 613.001.451. The work of JvL was further supported by the NWO Gravitation Networks grant 024.002.003, an NWO TOP-GO grant and by an ERC Starting Grant.

References

1. Albert, R., Jeong, H., Barabási, A.L.: Internet: Diameter of the world-wide web. Nature **401**(6749) (1999) 130–131
2. Boguñá, M., Pastor-Satorras, R.: Class of correlated random networks with hidden variables. Phys. Rev. E **68** (2003) 036112
3. Bollobás, B., Janson, S., Riordan, O.: The phase transition in inhomogeneous random graphs. Random Structures & Algorithms **31**(1) (2007) 3–122
4. Brach, P., Cygan, M., Łacki, J., Sankowski, P.: Algorithmic complexity of power law networks. In: Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '16, Philadelphia, PA, USA, Society for Industrial and Applied Mathematics (2016) 1306–1325
5. Britton, T., Deijfen, M., Martin-Löf, A.: Generating simple random graphs with prescribed degree distribution. J. Stat. Phys. **124**(6) (2006) 1377–1397
6. Chung, F., Lu, L.: The average distances in random graphs with given expected degrees. Proc. Natl. Acad. Sci. USA **99**(25) (2002) 15879–15882 (electronic)
7. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. SIAM Rev. **51**(4) (2009) 661–703
8. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. In: ACM SIGCOMM Computer Communication Review. Volume 29., ACM (1999) 251–262
9. Fountoulakis, N., Friedrich, T., Hermelin, D.: On the average-case complexity of parameterized clique. arXiv:1410.6400v1 (2014)
10. Fountoulakis, N., Friedrich, T., Hermelin, D.: On the average-case complexity of parameterized clique. Theoretical Computer Science **576** (apr 2015) 18–29
11. Friedrich, T., Krohmer, A.: Cliques in hyperbolic random graphs. In: INFOCOM proceedings 2015, IEEE (2015) 1544–1552
12. Friedrich, T., Krohmer, A.: Parameterized clique on inhomogeneous random graphs. Discrete Applied Mathematics **184** (mar 2015) 130–138
13. Garey, M.R., Johnson, D.S., Garey, M.R.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W H FREEMAN & CO (2011)
14. Grochow, J.A., Kellis, M.: Network motif discovery using subgraph enumeration and symmetry-breaking. In: In RECOMB. (2007) 92–106
15. Heydari, H., Taheri, S.M.: Distributed maximal independent set on inhomogeneous random graphs. In: 2017 2nd Conference on Swarm Intelligence and Evolutionary Computation (CSIEC), IEEE (mar 2017)
16. van der Hofstad, R.: Random Graphs and Complex Networks Vol. 1. Cambridge University Press (2017)
17. van der Hofstad, R., van Leeuwaarden, J.S.H., Stegehuis, C.: Optimal subgraph structures in scale-free networks. arXiv:1709.03466 (2017)

18. Janson, S., Łuczak, T., Norros, I.: Large cliques in a power-law random graph. Journal of Applied Probability **47**(04) (dec 2010) 1124–1135
19. Jeong, H., Tombor, B., Albert, R., Oltvai, Z.N., Barabási, A.L.: The large-scale organization of metabolic networks. Nature **407**(6804) (2000) 651–654
20. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of computer computations. Springer (1972) 85–103
21. Kashtan, N., Itzkovitz, S., Milo, R., Alon, U.: Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. Bioinformatics **20**(11) (2004) 1746–1758
22. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data` (2014) Date of access: 14/03/2017.
23. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi.me - weaving chinese linking open data. In: The Semantic Web – ISWC 2011. Springer Nature (2011) 205–220
24. Norros, I., Reittu, H.: On a conditionally poissonian graph process. Adv. Appl. Probab. **38**(01) (2006) 59–75
25. Omidi, S., Schreiber, F., Masoudi-Nejad, A.: MODA: An efficient algorithm for network motif discovery in biological networks. Genes & Genetic Systems **84**(5) (2009) 385–395
26. Park, J., Newman, M.E.J.: Statistical mechanics of networks. Phys. Rev. E **70** (2004) 066117
27. Schreiber, F., Schwobbermeyer, H.: MAVisto: a tool for the exploration of network motifs. Bioinformatics **21**(17) (jul 2005) 3572–3574
28. Vázquez, A., Pastor-Satorras, R., Vespignani, A.: Large-scale topological and dynamical properties of the internet. Phys. Rev. E **65** (2002) 066130
29. Williams, V.V., Wang, J.R., Williams, R., Yu, H.: Finding four-node subgraphs in triangle time. In: Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '15, Philadelphia, PA, USA, Society for Industrial and Applied Mathematics (2015) 1671–1680