Models and Algorithms for Graph Watermarking

David Eppstein[1], Michael T. Goodrich[1], Jenny Lam[2(✉)], Nil Mamano[1], Michael Mitzenmacher[3], and Manuel Torres[1]

[1] Department of Computer Science, University of California, Irvine, CA, USA
[2] Department of Computer Science, San José State University, San José, CA, USA
jenny.lam@sjsu.edu
[3] Department of Computer Science, Harvard University, Cambridge, MA, USA

Abstract. We introduce models and algorithmic foundations for graph watermarking. Our approach is based on characterizing the feasibility of graph watermarking in terms of keygen, marking, and identification functions defined over graph families with known distributions. We demonstrate the strength of this approach with exemplary watermarking schemes for two random graph models, the classic Erdős-Rényi model and a random power-law graph model, both of which are used to model real-world networks.

1 Introduction

In the classic media watermarking problem, we are given a digital representation, R, for some media object, O, such as a piece of music, a video, or an image, such that there is a rich space, $\mathcal{R}$, of possible representations for O besides R that are all more-or-less equivalent. Informally, a *digital watermarking* scheme for O is a function that maps R and a reasonably short random message, m, to an alternative representation, R', for O in $\mathcal{R}$. The verification of such a marking scheme takes R and a presumably-marked representation, R'' (which was possibly altered by an adversary), along with the set of messages previously used for marking, and it either identifies the message from this set that was assigned to R'' or it indicates a failure. Ideally, it should difficult for an adversary to transform a representation, R' (which he was given), into another representation R'' in $\mathcal{R}$, that causes the identification function to fail. Some example applications of such digital watermarking schemes include steganographic communication and marking digital works for copyright protection.

With respect to digital representations of media objects that are intended to be rendered for human performances, such as music, videos, and images, there is a well-established literature on digital watermarking schemes and even well-developed models for such schemes (e.g., see Hopper *et al.* [8]). Typically, such watermarking schemes take advantage of the fact that rendered works have many possible representations with almost imperceptibly different renderings from the perspective of a human viewer or listener.

In this paper, we are inspired by recent systems work on *graph watermarking* by Zhao *et al.* [18], who propose a digital watermarking scheme for graphs, such

M. Bishop and A.C.A. Nascimento (Eds.): ISC 2016, LNCS 9866, pp. 283–301, 2016.
DOI: 10.1007/978-3-319-45871-7_18

as social networks, protein-interaction graphs, etc., which are to be used for commercial, entertainment, or scientific purposes. This work by Zhao *et al.* presents a system and experimental results for their particular method for performing graph watermarking, but it is lacking in formal security and algorithmic foundations. For example, Zhao *et al.* do not provide formal proofs for circumstances under which graph watermarking is undetectable or when it is computationally feasible. Thus, as complementary work to the systems results of Zhao *et al.*, we are interested in the present paper in providing models and algorithms for graph watermarking, in the spirit of the watermarking model provided by Hopper *et al.* [8] for media files. In particular, we are interested in providing a framework for identifying when graph watermarking is secure and computationally feasible.

Additional Related Work. Under the term "graph watermarking," there is some additional work, although it is not actually for the problem of graph watermarking as we are defining it. For instance, there is a line of research involving software watermarking using graph-theoretic concepts and encodings. In this case, the object being marked is a piece of software and the goal of a "graph watermarking" scheme is to create a graph, G, from a message, m, and then embed G into the control flow of a piece of software, S, to mark S. Examples of such work include pioneering work by Collberg and Thomborson [6], as well as subsequent work by Venkatesan, Vazirani, and Sinha [16] and Collberg *et al.* [5]. This work on software watermarking differs from the graph watermarking problem we study in the present paper, however, because in the graph watermarking problem we study an input graph is provided and we want to alter it to add a mark. In the graph-based software watermarking problem, a graph is instead created from a message to have a specific, known structure, such as being a permutation graph, and then that graph is embedded into the control flow of the piece of software.

A line of research that is more related to the graph watermarking problem we study is anonymization and de-anonymization for social networks. One of the closest examples of such prior work is by Backstrom, Dwork, and Kleinberg [1], who show how to introduce a small set of "rogue" vertices into a social network and connect them to each other and to other vertices so that if that same network is approximately replicated in another setting it is easy to match the two copies. Such work differs from graph watermarking, however, because the set of rogue vertices are designed to "stand out" from the rest of the graph rather than "blend in," and it may in some cases be relatively easy for an adversary to identify and remove such rogue vertices. In addition to this work, also of note is work by Narayanan and Shmatikov [13], who study the problem of approximately matching two social networks without marking, as well as the work on Khanna and Zane [9] for watermarking road networks by perterbing vertex positions (which is a marking method outside the scope of our approach).

Our Results. In this paper, we introduce a general graph watermarking framework that is based on the use of key generation, marking, and identification functions, as well as a hypothetical watermarking security experiment (which would be performed by an adversary). We define these functions in terms of graphs taken over random families of graphs, which allows us to quantify situations in which graph watermarking is provably feasible.

We also provide some graph watermarking schemes as examples of our framework, defined in terms of the classic Erdős-Rényi random-graph model and a random power-law graph model. Our schemes extend and build upon previous results on graph isomorphism for these graph families, which may be of independent interest. In particular, we design simple marking schemes for these random graph families based on simple edge-flipping strategies involving high- and medium-degree vertices. Analyzing the correctness of our schemes is quite nontrivial, however, and our analysis and proofs involve intricate probabilistic arguments, some of which we include in the ePrint version of this paper [7]. We provide an analysis of our scheme against adversaries that can themselves flip edges in order to defeat our mark identification algorithms. In addition, we provide experimental validation of our algorithms, showing that our edge-flipping scheme can succeed for a graph without specific knowledge of the parameters of its deriving graph family.

2 Our Watermarking Framework

Suppose we are given an undirected graph, $G = (V, E)$, that we wish to mark. To define the security of a watermarking scheme for G, G must come from a family of graphs with some degree of entropy [19]. We formalize this by assuming a probability distribution $\mathcal{D}$ over the family $\mathcal{G}$ of graphs from which G is taken.

Definition 1. *A* graph watermarking scheme *is a tuple* (keygen, mark, identify) *over a set, $\mathcal{G}$, of graphs where*

- keygen : $\mathbb{N} \times \mathbb{N} \to$ Aux *is a private key generation function, such that* keygen(ℓ, n) *is a list of ℓ (pseudo-)random graph elements, such as vertices and/or vertex pairs, defined over a graph of n vertices. These candidate locations for marking are defined independent of a specific graph; that is, vertices in* Aux *are identified simply by the numbering from* 1 *to n. For example,* keygen(ℓ, n) *could be a small random graph, H, and some random edges to connect H to a larger input graph [19], or* keygen(ℓ, n) *could be a set of vertex pairs in an input graph that form candidate locations for marking.*
- mark : Aux $\times \mathcal{G} \to \mathbb{N} \times \mathcal{G}$ *takes a private key z generated by* keygen*, and a specific graph G from $\mathcal{G}$, and returns a pair, $S = (\mathsf{id}, H)$, such that* id *is a unique identifier for H and H is the graph obtained by adding the mark determined by* id *to G in the location determined by the private key z.* mark *is called every time a different marked copy needs to be produced, with the i-th copy being denoted by $S_i = (\mathsf{id}_i, H_i)$. Therefore, the unique identifiers should be thought of as being generated randomly. To associate a marked graph H_i with the user who receives it, the watermarking scheme can be augmented with a table storing user name and unique identifiers. Alternatively, the identifiers*

can be generated pseudo-randomly as a hash of a private key provided by the user.

– $\mathsf{identify} : \mathsf{Aux} \times \mathcal{G} \times \mathbb{N}^k \times \mathcal{G} \rightarrow \mathbb{N} \cup \{\perp\}$ *takes a private key from* Aux*, the original graph,* G*,* k *identifiers of previously-marked copies of* G*, and a test graph,* G'*, and it returns the identifier,* id_i*, of the watermarked graph that it is identifying as a match for* G'*. It may also return* $\perp$*, as an indication of failure, if it does not identify any of the graphs* H_i *as a match for* G'*.*

In addition, in order for a watermarking scheme to be effective, we require that with high probability[1] *over the graphs from* $\mathcal{G}$ *and* k *output pairs,* $S_1, \ldots, S_k$ *of* $\mathsf{mark}(z, G)$*, for any* $(\mathsf{id}, G') = S_i$*, we have* $\mathsf{identify}(z, G, \mathsf{id}_1, \ldots, \mathsf{id}_k, G') = \mathsf{id}$*.*

Algorithm 1 shows a hypothetical security experiment for a watermarking scheme with respect to an adversary, $A : \mathcal{G} \rightarrow \mathcal{G}$, who is trying to defeat the scheme. Intuitively, in the hypothetical experiment, we generate a key z, choose a graph G, from family $\mathcal{G}$ according to distribution $\mathcal{D}$ (as discussed above), and then generate k marked graphs according to our scheme (for some set of k messages). Next, we randomly choose one of the marked graphs, G', and communicate it to an adversary. The adversary then outputs a graph G_A that is similar to G' where his goal is to cause our identification algorithm to fail on G_A.

Algorithm 1. Hypothetical Watermarking Security Experiment

```
experiment($A, k, \ell, n$):
  1. $z \leftarrow \mathsf{keygen}(\ell, n)$
  2. $G \leftarrow_{\mathcal{D}} \mathcal{G}$
  3. $S_i \leftarrow \mathsf{mark}(z, G)$, for $i = 1, \ldots, k$
  4. randomly choose $S_i = (\mathsf{id}, G')$ from $\{S_1, \ldots, S_k\}$
  5. $G_A \leftarrow A(G')$
```

In order to characterize differences between graphs, we assume a similarity measure $\mathsf{dist} : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$, defining the distance between graphs in family $\mathcal{G}$. We also include a similarity threshold θ, that defines the advantage of an adversary performing the experiment in Algorithm 1. Specifically, the *advantage* of an adversary, $A : \mathcal{G} \rightarrow \mathcal{G}$ who is trying to defeat our watermarking scheme is

$$\mathbb{P}\left[\mathsf{dist}(G, G_A) < \theta \text{ and } \mathsf{identify}(z, G, \mathsf{id}_1, \ldots, \mathsf{id}_k, G_A) \neq \mathsf{id}\right].$$

The watermarking scheme is $(\mathcal{D}, \mathsf{dist}, \theta, k, \ell)$-secure against adversary A if the similarity threshold is θ and A's advantage is *polynomially negligible* (i.e., is $O(n^{-a})$ for some $a > 0$).

Examples of adversaries could include the following:

– *Arbitrary edge-flipping adversary*: a malicious adversary who can arbitrarily flip edges in the graph. That is, the adversary adds an edge if it is not already there, and removes it otherwise.

[1] Or "*whp*," that is, with probability at least $1 - O(n^{-a})$, for some $a > 0$.

- *Random edge-flipping adversary*: an adversary who independently flips each edge with a given probability.
- *Arbitrary adversary*: a malicious adversary who can arbitrarily add and/or remove vertices and flip edges in the graph.
- *Random adversary*: an adversary who independently adds and/or removes vertices with a given probability and independently flips each edge with a given probability.

Random Graph Models. As defined above, a graph watermarking scheme requires that graphs to be marked come from some distribution. In this paper, we consider two families of random graphs—the classic Erdős-Rényi model and a random power-law graph model—which should capture large classes of applications where graph watermarking would be of interest.

Definition 2 (The Erdős-Rényi model). *A* random graph $G(n, p)$ *is a graph with n vertices, where each of the $\binom{n}{2}$ possible edges appears in the graph independently with probability p.*

Definition 3 (The random power-law graph model, Sect. 5.3 of [4]). *Given a sequence $\mathbf{w} = (w_1, w_2, \ldots, w_n)$, such that* $\max_i w_i^2 < \sum_k w_k$*, the* general random graph $G(\mathbf{w})$ *is defined by labeling the vertices* 1 *through n and choosing each edge (i, j) independently from the others with probability $p[i, j] = \rho w_i w_j$, where $\rho = 1/\sum_j w_j$.*

We define a random power-law graph $G(\mathbf{w}^\gamma)$ *parameterized by the maximum degree m and average degree w. Let $w_i = ci^{-1/(\gamma-1)}$ for values of i in the range between i_0 and $i_0 + n$, where*

$$c = \frac{\gamma-2}{\gamma-1} w n^{\frac{1}{\gamma-1}}, \qquad i_0 = n\left(\frac{w(\gamma-2)}{m(\gamma-1)}\right)^{\gamma-1}. \tag{1}$$

This definition implies that each edge (i, j) appears with probability

$$P[i,j] = K_0 \left(n^{\gamma-3} ij\right)^{-\frac{1}{\gamma-1}}, \quad \textit{where } K_0 \overset{def}{=} \left(\frac{\gamma-2}{\gamma-1}\right)^2 w. \tag{2}$$

Graph Watermarking Algorithms. We discuss some instantiations of the graph watermarking framework defined above. Unlike previous watermarking or de-anonymization schemes that add vertices [1,19], we describe an effective and efficient scheme based solely on edge flipping. Such an approach would be especially useful for applications where it could be infeasible to add vertices as part of a watermark.

Our scheme does not require adding labels to the vertices or additional objects stored in the graph for identification purposes. Instead, we simply rely on the structural properties of graphs for the purposes of marking. In particular, we focus on the use of vertex degrees, that is, the number of edges incident on each vertex. We identify high and medium degree vertices as candidates for finding edges that can be flipped in the course of marking. The specific degree thresholds for what we mean by "high-degree" and "medium-degree" depend on the graph family, however, so we postpone defining these notions precisely until our analysis sections.

Algorithms providing an example implementation of our graph watermarking scheme are shown in Algorithm 2. The keygen algorithm randomly selects a set of candidate vertex pairs for flipping, from among the high- and medium-degree vertices, with no vertex being incident to more than a parameter t of candidate pairs. We introduce a procedure, $\mathsf{label}(G)$, which labels high-degree vertices by their degree ranks and each medium-degree vertex, w, by a bit vector identifying its high-degree adjacencies. This bit vector has a bit for each high-degree vertex, which is 1 for neighbors of w and 0 for non-neighbors. The algorithm $\mathsf{mark}(z, G)$, takes a random set of candidate edges and a graph, G, and it flips the corresponding edges in G according to a resampling of the edges using the distribution $\mathcal{D}$. The algorithm, $\mathsf{approximate\text{-}isomorphism}(G, H)$, returns a mapping of the high- and medium-degree vertices in G to matching high- and medium-degree vertices in H, if possible. The algorithm, $\mathsf{identify}(z, G, \mathsf{id}_1, \ldots, \mathsf{id}_k, H)$, uses the approximate isomorphism algorithm to match up high- and medium-degree vertices in G and H, and then it extracts the bit-vector from this matching using z.

As mentioned above, we also need a notion of distance for graphs. We use two different such notions. The first is the graph edit distance, which is the minimum number of edges needed to flip to go from one graph to another. The second is vertex distance, which intuitively is an edge-flipping metric localized to vertices.

Definition 4 (Graph distances). *Let $\mathcal{G}$ be the set of graphs on n vertices. If $G, H \in \mathcal{G}$, define Π as the set of bijections between the vertex sets $V(G)$ and $V(H)$. Define the* graph edit distance $\mathsf{dist}_e : \mathcal{G} \times \mathcal{G} \to \mathbb{N}$ *as*

$$\mathsf{dist}_e(G, H) = \min_{\pi \in \Pi} |E(G) \oplus_\pi E(H)|,$$

where $\oplus_\pi$ is the symmetric difference of the two edge sets under correspondence π. Define the vertex distance $\mathsf{dist}_v : \mathcal{G} \times \mathcal{G} \to \mathbb{N}$ *as*

$$\mathsf{dist}_v(G, H) = \min_{\pi \in \Pi} \max_{v \in V(G)} |E(v) \oplus_\pi E(\pi(v))|,$$

where $E(v)$ is the set of edges incident to v.

3 Identifying High- and Medium-Degree Vertices

We begin analyzing our proposed graph watermarking scheme by showing how high- and medium-degree vertices can be identified under our two random graph distributions. We ignore low-degree vertices: their information content and distinguishability are low, and they are not used by our example scheme.

We first find a threshold number k such that the k vertices with highest degree are likely to have distinct and well-separated degree values. We call these k

Algorithm 2. Watermarking scheme for random graphs.

t: the maximum number of flipped edges that can be adjacent to the same vertex. keygen(ℓ, n):

```
1. Let $x$ denote the total number of high- and medium-degree vertices
2. $X = \{(u, v) \mid 1 \leq u < v \leq x\}$
3. Let $z$ be a list of $\ell$ pairs randomly sampled (without replacement) from $X$ such
   that no end vertex appears more than $t$ times
4. return $z$
```

label(G):

```
1. sort the vertices in decreasing order by degree and identify the high- and
   medium-degree vertices
2. if the degrees of high-degree vertices are not unique, return failure
3. label each high-degree vertex with its position in the vertex sequence
4. label each medium-degree vertex with a bit vector encoding its high-degree
   adjacencies
5. if the bit vectors are not unique, return failure
6. otherwise, return the labelings
```

mark(z, G):

```
 1. $S = \varnothing$
 2. $V$ is the set of high- and medium-degree vertices of $G$, sorted lexicographically by
    their labels given by $L = \mathsf{label}(G)$
 3. generate an $\ell$-bit string id where each bit $i$ is independently set to 1 with
    probability $p_{z[i]}$, where $p_{z[i]}$ is the probability of the edge $z[i]$ in $\mathcal{D}$
 4. let $H$ be a copy of $G$
 5. for $j$ from 1 to $\ell$:
 6.    $(u, v) = z[j]$
 7.    if id$[j]$ is 1:
 8.       insert edge $(V[u], V[v])$ in $H$
 9.    else:
10.       remove edge $(V[u], V[v])$ from $H$
11. return (id, $H$)
```

approximate-isomorphism(G, H):

```
1. call label$(G)$ and label$(H)$, returning failure if either of these fail.
2. match each of $G$'s high-degree vertices with the vertex in $H$ with the same label.
3. match each of $G$'s medium-degree vertices with the vertex in $H$ whose label is
   closest in Hamming distance.
4. if $H$ has a vertex that is matched more than once, return failure.
5. otherwise, return the (partial) vertex assignments between $G$ and $H$.
```

identify$(z, G, \mathsf{id}_1, \ldots, \mathsf{id}_k, H)$:

```
1. find an approximate-isomorphism$(G, H)$, returning $\perp$ if failure occurred at any step.
2. $V$ is the set of high- and medium-degree vertices of $G$, sorted lexicographically by
   their labels given by $L = \mathsf{label}(G)$
3. $V'$ is the set of vertices of $H$ identified as corresponding to those in $V$, in that
   same order.
4. id is an empty bit string
5. for $(u, v)$ in $z$ (from left to right):
6.    $b = 1$ iff there is an edge between $V'[u]$ and $V'[v]$ in $H$.
7.    append $b$ to id
8. return among the id$_i$'s the one closest to id
```

vertices the *high-degree* vertices. Next, we look among the remaining vertices for those that are well-separated in terms of their high-degree neighbors. Specifically, the (high-degree) *neighborhood distance* between two vertices is the number of high-degree vertices which are connected to exactly one of the two vertices. Note that we will omit the term "high-degree" in "high-degree neighborhood distance" from now on, as it will always be implied.

In the Erdős-Rényi model, we show that all vertices that are not high-degree nevertheless have well-separated high-degree neighborhoods whp. In the random power-law graph model, however, there will be many lower-degree vertices whose high-degree neighborhoods cannot be separated. Those that have well-separated high-degree neighborhoods with high probability form the medium-degree vertices, and the rest are the low-degree vertices.

For completeness, we include the following well-known Chernoff concentration bound, which we will refer to time and again.

Lemma 5 (Chernoff inequality [4]). *Let $X_1, \ldots, X_n$ be independent random variables with*

$$\mathbb{P}[X_i = 1] = p_i, \qquad \mathbb{P}[X_i = 0] = 1 - p_i.$$

We consider the sum $X = \sum_{i=1}^{n} X_i$, with expectation $\mathbb{E}[X] = \sum_{i=1}^{n} p_i$. Then

$$\mathbb{P}[X \leq \mathbb{E}[X] - \lambda] \leq e^{-\frac{\lambda^2}{2\mathbb{E}[X]}},$$
$$\mathbb{P}[X \geq \mathbb{E}[X] + \lambda] \leq e^{-\frac{\lambda^2}{2\mathbb{E}[X]+\lambda/3}}.$$

Vertex Separation in the Erdős-Rényi Model. Let us next consider vertex separation results for the classic Erdős-Rényi random-graph model. Recall that in this model, each edge is chosen independently with probability p.

Definition 6. *Index vertices in non-increasing order by degree. Let d_i represent the i-th highest degree in the graph. Given $h = O(n)$, we say that a vertex is* high-degree *with respect to d_h if it has degree at least d_h. Otherwise, we say that the vertex is* medium-degree*.*

Note that in this random-graph model, there are no low-degree vertices.

Definition 7. *A graph is (d, d')-*separated *if all high-degree vertices differ in their degree by at least d and all medium-degree vertices are neighborhood distance d' apart.*

Note: this definition depends on how high-degree or medium-degree vertices are defined and will therefore be different for the random power-law graph model.

Lemma 8 (Extension of Theorem 3.15 in [2]). *Suppose $m = o(pqn/\log n)^{1/4}$, $m \to \infty$, and $\alpha(n) \to 0$. Then with probability*

$$1 - m\alpha(n) - 1/\left[m\left(\log(n/m)\right)^2\right],$$

$G(n,p)$ is such that

$$d_i - d_{i+1} \geq \frac{\alpha(n)}{m^2}\left(\frac{pqn}{\log n}\right)^{1/2} \text{ for every } i < m,$$

where $q = 1 - p$.

Proof. See the ePrint version [7]. □

Lemma 9 (Vertex separation in the Erdős-Rényi model). *Let $0 < \varepsilon < 1/9$, $d \geq 3$, $C \geq 3$, $h = n^{(1-\varepsilon)/8}$. Suppose $0 < p = p(n) \leq \frac{1}{2}$ is such that $p = \omega(n^{-\varepsilon}\log n)$. Then $G(n,p)$ is $(d, C\log n)$-separated with probability $1 - O(n^{-(1-\varepsilon)/8})$.*

Proof. See the ePrint version [7]. □

Thus, high-degree vertices are well-separated with high probability in the Erdős-Rényi model, and the medium-degree vertices are distinguished with high probability by their high-degree neighborhoods.

Vertex Separation in the Random Power-Law Graph Model. We next study vertex separation for a random power-law graph model, which can match the degree distributions of many graphs that naturally occur in social networking and science. For more information about power-law graphs and their applications, see e.g. [3,12,14].

In the random power-law graph model, vertex indices are used to define edge weights and therefore do not necessarily start at 1. The lowest index that corresponds to an actual vertex is denoted i_0. So vertex indices range from i_0 to $i_0 + n$. Additionally, there are two other special indices i_H and i_M, which we define in this section, that separate the three classes of vertices.

Definition 10. *The vertices ranging from i_0 to i_H are the* high-degree *vertices, those that range from $i_H + 1$ to i_M are the* medium-degree *vertices, and those beyond i_M are the* low-degree *vertices.*

In this model, the value of i_0 is constrained by the requirement that $P[i_0, i_0] < 1$. When $\gamma \geq 3$, this constraint is not actually restrictive. However, when $\gamma < 3$, i_0 must be asymptotically greater than $n^{-(\gamma-3)/2}$. The constraints on i_0 also constrain the value of the maximal and average degree of the graph.

We define i_H and i_M to be independent of i_0, but dependent on parameters that control the amount and probability of separation at each level. The constraints that $i_0 < i_H$ and $i_H < i_M$ translate into corresponding restrictions on the valid values of γ, namely that $\gamma > 5/2$ and $\gamma < 3$. We define i_H in the following lemma.

Lemma 11 (Separation of high-degree vertices). *In the $G(\mathbf{w}^\gamma)$ model, let $\delta_i = |w_{i+1} - w_i|/2$. Then,*

$$\frac{c}{2(\gamma-1)}(i+1)^{-\frac{\gamma}{\gamma-1}} \leq \delta_i \leq \frac{c}{2(\gamma-1)} i^{-\frac{\gamma}{\gamma-1}}. \tag{3}$$

Moreover, for all ε_1 *satisfying* $0 < \varepsilon_1 \leq 1$ *and* $C_1 > 0$, *the probability that*

$$|\deg(i) - w_i| < \varepsilon_1 \delta_i \qquad \text{for all } i \leq i_H \overset{def}{=} \left(\frac{c\varepsilon_1^2}{16(\gamma-1)^2 C_1 \log n}\right)^{\frac{\gamma-1}{2\gamma-1}}$$

is at least $1 - n^{-C_1}$.

Proof. The first statement follows from the fact that w_i is a convex function of i and from taking its derivative at i and $i+1$.

For the second statement, let $C > 0$ and let $i'_H \overset{\text{def}}{=} \left(\frac{c\varepsilon_1^2}{8(\gamma-1)^2 C \log n}\right)^{\frac{\gamma-1}{2\gamma-1}}$. We will show that if $i \leq i'_H$, then

$$\mathbb{P}\left[|\deg(i) - w_i| \geq \varepsilon_1 \delta_i\right] < n^{-C}. \tag{4}$$

Now we choose C such that $C_1 + \log i_H / \log n < C \leq 2C_1$. The inequality $C \leq 2C_1$ implies that $i_H \leq i'_H$ and (4) holds for all $i \leq i_H$. By the union bound applied to (4)

$$\mathbb{P}\left[\exists i \leq i_H, |\deg(i) - w_i| \geq \varepsilon_1 \delta_i\right] \leq i_H n^{-C}.$$

Since $C_1 + \log i_H / \log n < C$, the right hand side is bounded above by n^{-C_1}. This proves the result.

Now, we prove (4). Clearly, since $\delta_i = (w_i - w_{i+1})/2$, we have that $w_i \geq \delta_i$. So if $\varepsilon_1 \leq 1$ and $\lambda_i = \varepsilon_1 \delta_i$, then $w_i \geq \lambda_i / 3$. This implies that

$$\frac{\lambda_i^2}{w_i + \lambda_i/3} \geq \frac{\lambda_i^2}{2w_i} \geq \frac{c\varepsilon_1^2}{8(\gamma-1)^2} i^{-\frac{2\gamma-1}{\gamma-1}},$$

where the second inequality follows from (3) and the definition of w_i given in Definition 3. If $i \leq i'_H$, the right hand side is lower-bounded by $C \log n$. The result follows by applying a Chernoff bound (Lemma 5). □

For simplicity, we often use the following observation.

Observation 12. *Rewriting* i_H *to show its dependence on* n, *we have*

$$i_H(\varepsilon_1, C_1) = K_1(\varepsilon_1, C_1)\ n^{\frac{1}{2\gamma-1}} (\log n)^{-\frac{\gamma-1}{2\gamma-1}}, \quad K_1(\varepsilon_1, C_1) \overset{def}{=} \left(\frac{\gamma-2}{(\gamma-1)^3} \frac{w\varepsilon_1^2}{16C_1}\right)^{\frac{\gamma-1}{2\gamma-1}}. \tag{5}$$

For the graph model to make sense, the high-degree threshold must be asymptotically greater than the lowest index. In other words, we must have that $i_0 = o(i_H)$. *Since* $i_0 = \Omega(n^{-(\gamma-3)/2})$, *this implies that* $\gamma > 5/2$.

We next define i_M, the degree threshold for medium-degree vertices, in the following lemma.

Lemma 13 (Separation of medium-degree vertices). *Let* K_0 *be defined as in Definition 3,* $K_1(\varepsilon_1, C_1)$ *be defined as in (5), and*

$$K_2(\varepsilon_1, C_1, \varepsilon_2, C_2) \overset{def}{=} \frac{K_0^{\gamma-1} K_1^{\gamma-2}(\varepsilon_1, C_1)}{(C_2 + 2\Gamma + 2\log(K_0^{\gamma-1} K_1^{\gamma-2}(\varepsilon_1, C_1)) + 2\varepsilon_2)^{\gamma-1}}. \tag{6}$$

Let X_{ij} denote the neighborhood distance between two vertices i and j in $G(\mathbf{w}^{\gamma})$. If $5/2 < \gamma < 3$, for every $\varepsilon_2 > 0$ and $C_2 > 0$, the probability that

$$X_{ij} > \varepsilon_2 \log n, \quad \textit{for all } i_H \le i, j \le i_M$$

where

$$i_M(\varepsilon_1, C_1, \varepsilon_2, C_2) \overset{def}{=} K_2(\varepsilon_1, C_1, \varepsilon_2, C_2)\ n^{\Gamma}\,(\log n)^{-\frac{3(\gamma-1)^2}{2\gamma-1}}, \quad \Gamma \overset{def}{=} -\frac{2\gamma^2 - 8\gamma + 5}{2\gamma - 1}, \tag{7}$$

is at least $1 - n^{-C_2}$ for sufficiently large n.

Proof. Let $C > 0$ and let

$$i'_M \overset{\text{def}}{=} \left(\frac{C_2 + 2\Gamma + 2\log(K_0^{\gamma-1} K_1^{\gamma-2}) + 2\varepsilon_2}{C + 2\varepsilon_2} \right)^{\gamma-1} i_M.$$

We claim that if $i_H \le i, j \le i'_M$, then

$$\mathbb{P}\left[X_{ij} \le \varepsilon_2 \log n\right] \le n^{-C}. \tag{8}$$

If we choose $C = C_2 + 2\Gamma + 2\log K_0^{\gamma-1} K_1^{\gamma-2}$, we have that $i_M = i'_M$, so that (8) applies to all i, j such that $i, j \le i_M$. Moreover, since

$$i_M \le K_0^{\gamma-1} K_1^{\gamma-2} n^{\Gamma} \le n^{\log(K_0^{\gamma-1} K_1^{\gamma-2})} n^{\Gamma},$$

our choice of C implies that $i_M^2\ n^{-C} \le n^{-C_2}$. By applying the union bound to (8), we have

$$\mathbb{P}\left[\exists i, j \text{ s.t. } i_H \le i, j \le i_M,\ X_{ij} \le\ \varepsilon_2 \log n\right]\ \le\ i_M^2 n^{-C}\ \le\ n^{-C_2},$$

which establishes the lemma.

Let us now prove the claim. Observe that X_{ij} is the sum over the high-degree vertices k, of indicator variables X_{ij}^k for the event that vertex k is connected to exactly one of the vertices i and j. It i For fixed i and j, these are independent random variables. Therefore, we can apply a Chernoff bound. The probability that $X_{ij}^k = 1$ is

$$P[i,k](1 - P[j,k]) + P[j,k](1 - P[i,k]) \ge 2P[i_M, i_H](1 - P[i_0, i_H]).$$

Since $P[i_0, i_H] \to 0$, for sufficiently large n, this expression is bounded below by $P[i_M, i_H]$, and

$$\mathbb{E}\left[X_{ij}\right] \ge i_H P[i_M, i_H] \ge (C + 2\varepsilon_2)\log n,$$

by (2), (5) and (7), as can be shown by a straightforward but lengthy computation. Let $d = \varepsilon_2 \log n$. This implies that

$$\frac{(\mathbb{E}\left[X_{ij}\right] - d)^2}{\mathbb{E}\left[X_{ij}\right]} \ge \mathbb{E}\left[X_{ij}\right] - 2d \ge C \log n.$$

Therefore, applying the Chernoff bound (Lemma 5) to the X_{ij}^k for fixed i and j and all high-degree vertices k proves the claim. □

Observation 14. *We would have the undesirable situation that* $i_M = o(1)$ *whenever* $\frac{2\gamma^2-8\gamma+5}{2\gamma-1} > 0$*, or equivalently when* $\gamma > 2 + \sqrt{3/2} > 3$*. In fact, in order for* $i_H = o(i_M)$*, we must have* $\gamma < 3$*.*

We illustrate the breakpoints for high-, medium-, and low-degree vertices in Fig. 1.

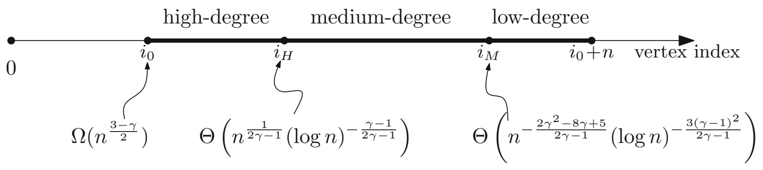
high-degree
medium-degree
low-degree
0
i_0
i_H
i_M
i_0+n
vertex index
$\Omega(n^{\frac{3-\gamma}{2}})$
$\Theta\left(n^{\frac{1}{2\gamma-1}}(\log n)^{-\frac{\gamma-1}{2\gamma-1}}\right)$
$\Theta\left(n^{-\frac{2\gamma^2-8\gamma+5}{2\gamma-1}}(\log n)^{-\frac{3(\gamma-1)^2}{2\gamma-1}}\right)$

Fig. 1. Degree breakpoints for the random power-law graph model.

The next lemma summarizes the above discussion and provides the forms of i_H and i_M that we use in our analysis.

Lemma 15 (Vertex separation in the power-law model). *Let* $5/2 < \gamma < 3$*. Fix* $\varepsilon > 0, C_1 > 0, C_2 > 0$*. Let* $i_H = i_H(\varepsilon_1, C_1)$ *and* $i_M = i_M(\varepsilon_1, C_1, \varepsilon_2, C_2)$ *where* $\varepsilon_1 = 1$ *and* $\varepsilon_2 = \varepsilon$*. Let*

$$d = n^{\frac{1}{2\gamma-1}} \qquad \text{and} \qquad d' = \log n.$$

For sufficiently large n*, the probability that a graph* $G(\mathbf{w}^\gamma)$ *is not* $(\varepsilon d, \varepsilon d')$*-separated is at most* $n^{-C_1} + n^{-C_2}$*.*

Proof. Let δ_i be defined as in Lemma 11. A straightforward computation using (1), (3), and (5) shows that

$$\delta_{i_H} \geq \text{constant} \cdot n^{\frac{1}{2\gamma-1}} (\log n)^{\frac{\gamma}{2\gamma-1}}.$$

So for sufficiently large n, we have $\delta_{i_H} \geq 3\varepsilon d/2$. For all $i \leq i_H$, the average degrees w_i of consecutive vertices are at least $3\varepsilon d/2$ apart. So for two high-degree vertices to be within εd of each other, at least one of the two must have degree at least $(3\varepsilon/2 - \varepsilon/2)d$ away from its expected degree. By Lemma 11, the probability that some high-degree vertex i satisfies $|\deg(i) - w_i| > \delta_{i_H}$ is at most n^{-C_1}.

By Lemma 13, the probability that there are two medium-degree vertices with neighborhood distance less than $\varepsilon d'$ is at most n^{-C_2}. □

Thus, our marking scheme for the random power-law graph model is effective.

4 Adversary Tolerance

In this section, we study the degree to which our exemplary graph watermarking scheme can tolerate an arbitrary edge-flipping adversary.

Theorem 16 (Security against an arbitrary edge-flipping adversary in the Erdős-Rényi model). *Let $0 < \varepsilon < 1/9$, $d \geq 3$, $h = n^{(1-\varepsilon)/8}$ and $p \leq 1/2$ such that $p = \omega(n^{-\varepsilon} \log n)$. Let d be sufficiently large so that*

$$\varepsilon \, \frac{d+1}{d-1} < 1. \tag{9}$$

Suppose the similarity measure is the vertex distance dist_v, the similarity threshold is $\theta = d$, we have a number $k = n^C$ of watermarked copies, and their identifiers are generated using $\ell = 8(2C + C')n^{\varepsilon}$ bits. Suppose also that the identifiers map to sets of edges of a graph constrained by the fact that no more than $t = d$ edges can be incident to any vertex. The watermarking scheme defined in Algorithm 2 is $(G(n,p), \mathsf{dist}_v, \theta, k, \ell)$-secure against any deterministic adversary.

The proof of this theorem relies on two lemmas. Lemma 17 identifies conditions under which a set of bit vectors with bits independently set to 1 is unlikely to have two close bit vectors. Lemma 18 states that a deterministic adversary's ability to guess the location of the watermark is limited. Informally, this is because the watermarked graph was obtained through a random process, so that there are many likely original graphs that could have produced it.

Lemma 17 (Separation of IDs). *Consider $k = n^C$ random bit strings of length ℓ, where each bit is independently set to 1, and the i-th bit is 1 with probability q_i satisfying $p \leq q_i \leq 1/2$ for a fixed value p. The probability that at least two of these strings are within Hamming distance $D = 4(2C + C') \log n$ of each other is at most $n^{-C'}$ if $\ell p \geq 2D$.*

Proof. See the ePrint version [7]. □

Lemma 18 (Guessing power of adversary). *Consider a complete graph on N vertices, and let r of its edges be red. Let s be a sample of ℓ edges chosen uniformly at random among those that satisfy the constraint that no more than t edges of the sample can be incident to any one vertex. Suppose also that ℓ, N and t are non-decreasing functions of n such that*

$$\frac{\ell^{t+1}}{N^{t-1}} \to 0 \text{ as } n \to \infty. \tag{10}$$

For sufficiently large N, the probability that s contains at least $R = 8\ell r/N^2$ red edges is bounded by $4 \exp\left(-12\ell r/(7N^2)\right)$. Moreover, if $\ell r/N^2 \to 0$, then the probability that s contains at least $R = 1$ red edge is bounded by $4 \exp\left(-cN^2/(\ell r)\right)$, for some $c > 0$ and for sufficiently large N.

Proof. See the ePrint version [7]. □

Proof (Theorem 16). An upper bound on the advantage of any deterministic adversary $A : \mathcal{G} \to \mathcal{G}$ on graphs on n vertices is given by the conditional probability

$$\mathbb{P}\left[\mathsf{identify}(z, G, \mathsf{id}_1, \ldots, \mathsf{id}_k, G_A) \neq \mathsf{id} | \mathsf{dist}_v(G, G_A) < \theta\right],$$

where the parameters passed to identify are defined according to the experiment in Algorithm 1. We show that this quantity is polynomially negligible.

For G_A to be successfully identified, it is sufficient for the following three conditions to hold:

1. the original graph $G = G(n, p)$ is $(4d, 4d)$-separated;
2. the Hamming distance between any two id and id′ involved in a pair in S is at least $D = 4(2C + C') \log n$;
3. A changes no edges of the watermark.

These are sufficient conditions because we only test graphs whose vertices had at most d incident edges modified by the adversary, and another d incident edges modified by the watermarking. So for original graphs that are $(4d, 4d)$-separated, the labeling of the vertices can be successfully recovered. Finally, if the adversary does not modify any potential edge that is part of the watermark, the id of the graph is intact and can be recovered from the labeling.

Now, by Lemma 9, the probability that $G(n, p)$ is not $(4d, 4d)$-separated is less than $O(n^{-(1-\varepsilon)/8})$. Moreover, since $\ell p \geq 2D$, by Lemma 17, the probability that there are two identifiers in S that are within D of each other is at most $n^{-C'}$.

Finally, for graphs in which an adversary makes fewer than d modifications per vertex, the total number of edges the adversary can modify is $r \leq dn/2$. Since all vertices are high- and medium-degree vertices in this model, $N = n$. Therefore, $\ell r/N^2 = O(1/n^{(1-\varepsilon)}) \to 0$. Eq. (9) guarantees that the hypothesis given by (10) of Lemma 18 is satisfied. Consequently, the probability that A changes one or more adversary edges is $O(\exp[cn^{1-\varepsilon}])$ for some constant c.

This proves that each of the three conditions listed above fails with polynomially negligible probability, which implies that the conditional probability is also polynomially negligible. □

Theorem 19 (Security against an arbitrary edge-flipping adversary in the random power-law graph model). *Let $5/2 < \gamma < 3$, $C > 0$, $i_H = i_H(\varepsilon_1, C_1)$ and $i_M = i_M(\varepsilon_1, C_1, \varepsilon_2, C_2)$ where $\varepsilon_1 = 1$, $\varepsilon_2 = 8(C + 1)$ and $C_1 = C_2 = C$.*

Let $p = P[i_M, i_M]$. Suppose the similarity measure is a vector of distances $\mathsf{dist} = (\mathsf{dist}_e, \mathsf{dist}_v)$, that the corresponding similarity threshold is the vector $\theta = (r, \log n)$ where $r = p(i_M)^2/32$ is the maximum number of edges the adversary can flip in total, and $\log n$ the maximum number of edges it can flip per vertex. Suppose that we have $k = n^{C''}$ watermarked copies of the graph, that we use $\ell = 8(2C'' + C')(\log n)/p$ to watermark a graph.

Suppose also that the identifiers map to sets of edges of a graph constrained by the fact that no more than $t = \log n$ edges can be incident to any vertex. Then the watermarking scheme defined in Algorithm 2 is $(G(\mathbf{w}^\gamma), \mathsf{dist} = (\mathsf{dist}_e, \mathsf{dist}_v), \theta = (r, \log n), k, \ell)$-secure against any deterministic adversary.

Proof. See the ePrint version [7]. □

Discussion. It is interesting to note how the differences in the two random graph models translate into differences in their watermarking schemes. The Erdős-Rényi model, with its uniform edge probability, allows for constant separation of high-degree vertices, at best. But all the vertices tend to be well-separated. On the other hand, the skewed edge distribution that is characteristic of the random power-law model allows high-degree vertices to be very well-separated, but a significant number of vertices—the low-degree ones, will not be easily distinguished.

These differences lead to the intuition that virtually all edges in the Erdős-Rényi model are candidates for use in a watermark, as long as only a constant number of selected edges are incident to any single vertex. Therefore, both our watermarking function and the adversary are allowed an approximately linear number of changes to the graph. Theorem 16 confirms this intuition with a scheme that proposes $O(n^{\varepsilon})$ bits for the watermark, and a nearly linear number $O(n)$ bits that the adversary may modify.

In contrast, the number of edges that can be used as part of a watermark in the random power-law graph model is limited by the number of distinguishable vertices, which is on the order of i_M or $O(n^{\varepsilon})$, where $\varepsilon = -\frac{2\gamma^2-8\gamma+5}{2\gamma-1}$.

5 Experiments

Although our paper is a foundational complement to the systems work of Zhao *et al.* [18], we nevertheless provide in this section the results of a small set of empirical tests of our methods, so as to experimentally reproduce the hypothetical watermarking security experiment from Algorithm 1. Our experiments are performed on two large social network graphs, Youtube [17] from the SNAP library [10], and Flickr [11], as well as a randomly generated graph drawn from the random power-law graph model distribution. Table 1 illustrates the basic properties of the networks. To generate the random power-law graph, we set the number of nodes to $n = 10000$, the maximum degree to $m = 1000$, the average degree to $w = 20$, and $\gamma = 2.75$.

To adapt our theoretical framework to the rough-and-tumble world of empirical realities, we made three modifications to our framework for the sake of our empirical tests.

Table 1. Network statistics

Network	# nodes	# edges	Max. degree	Avg. degree	Unique degree	Estimated γ
Power-law	10,000	94,431	960	18.89	14	—
Youtube	1,134,890	2,987,624	28,754	5.27	29	1.48
Flickr	1,715,256	15,554,181	27,203	18.14	130	1.62

Table 2. Experiment parameters

Network	# high-degree	# medium-degree	Key size	Marking dK-2 deviation
Power law	64	374	219	0.065
Youtube	256	113	184	0.033
Flickr	300	5901	3250	0.002

First, instead of using the high-degree and medium-degree thresholds derived from Lemmas 11 and 13, for the power-law distribution, to define the cutoffs for high-degree and medium-degree vertices, we used these and the other lemmas given above as justifications for the existence of such distinguishing sets of vertices and we then optimized the number of high- and medium-degree vertices to be values that work best in practice. The column, “Unique degree,” from Table 1 shows, for each network, the number of consecutive nodes with unique degree when considering the nodes in descending order of degree. Since this value is too small in most cases, we applied the principles of Lemmas 9 and 11 again, in a second-order fashion, to distinguish and order the high-degree nodes. In particular, in addition to the degree of each high-degree vertex, we also label each vertex with the list of degrees of its neighbors, sorted in decreasing order. With this change, we are not restricted in our choice of number of high-degree nodes as required by applying these lemmas only in a first-order fashion. Table 2 shows the values used in our experiments based on this second-order application. As medium-degree vertices, we picked the maximum number such that there are no collisions among their bit vectors of high-degree node adjacencies.

Second, instead of returning failure if (a) two high-degree nodes have the same degree and list of degrees of their neighbors, (b) two medium-degree nodes have the same bit vector, or (c) the approximate isomorphism is not injective, we instead proceed with the algorithm. Despite the existence of collisions, the remaining nodes often provide enough information to conclude successfully.

Finally, we simplified how we resampled (and flipped) edges in order to create a graph watermark, using our approach for the Erdős-Rényi model even for power-law graphs, since resampling uniformly among our small set of marked edges is likely not to cause major deviations in the graph’s distribution and, in any case, it is empirically difficult to determine the value of γ for real-world social networks. Therefore, we set the resampling probability to 0.5 so that it is consistent with the Erdős-Rényi model and so that each bit in the message is represented uniformly and independently.

Experiment Parameters. For the experiment parameters other than the original network and the number of high- and medium-degree nodes, we set the following values.

Maximum Flips Adjacent to Any Given Node During Marking: 1.

Key size: We set this to the maximum possible value (i.e., the number of high- and medium-degree vertices divided by two, as shown in Table 2), because the numbers of high- and medium- degree nodes are not large. This effectively means that every high- and medium-degree node has exactly one edge added or removed.

Number of Marked Graphs: 10.

Adversary: We used a time-efficient variation of the *arbitrary edge-flipping adversary*. This adversary selects a set of pairs of nodes randomly, and flips the potential edge among each pair.

Results. We evaluated how much distortion the adversary can introduce before our method fails to identify the leaked network correctly. For this purpose, we compared the identification success rate to the amount of distortion under different fractions of modified edges by the adversary. To estimate the success rate, we ran the experiment 10 times and reported the fraction of times that the leaked network was identified correctly. As a measure of distortion, we used the dK-2 deviation [18] between the original network and the version modified by

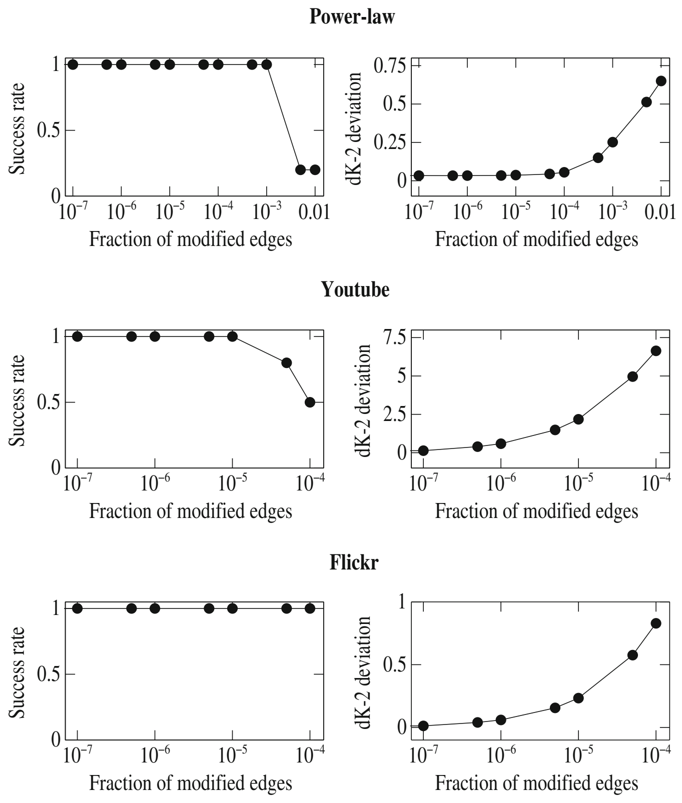

Fig. 2. Success rate and dK-2 deviation under different fractions of modified potential edges by the adversary, for the Power law, Youtube, and Flickr networks.

the adversary. The dK-2 deviation is the euclidean distance between the dK-2 series [15] of the two graphs, normalized by the number of tuples in the dK-2 series. The dK-2 deviation captures the differences between the joint degree distributions of the networks, that is, the probability that a randomly selected edge has as endpoints nodes with certain degrees. We average the dK-2 deviation among the 10 runs. Figure 2 shows the outcome of our experiments. Moreover, Table 2 shows the dK-2 deviation introduced by the marking alone.

Based on our experiments, the success rate of our scheme is high but it drops after a certain threshold. This demonstrates that there is a distinct range of adversarial edge flips that can be tolerated by our scheme. Specifically, our scheme worked well when the fraction of potential edges flipped by the adversary is up to 10^{-3} and 10^{-5} for the random power-law and Youtube networks, respectively. For these graphs, this number of flipped potential edges corresponds to 52.9 % and 215.6 % of the number of edges in the original graphs, respectively. For the Flickr network, the runtime of the adversary modification became excessive before the success rate could decrease, at a fraction of 10^{-4} of potential edges flipped.

The distortion introduced by the watermark is negligible compared to the distortion caused by the number of flips that the scheme can tolerate. On average, the marking modifies half of the edges on the key, which corresponds to $1.1 \cdot 10^{-3}, 3 \cdot 10^{-5}$, and 10^{-4} of the number of edges in the original random power-law, Youtube, and Flickr networks, respectively.

For the same number of flips, the dK-2 deviation in the Youtube network was much larger than in the Flickr network, which in turn was larger than that of the random power-law network. A possible explanation for this is that any set of uniform edge flips has a bigger effect on the dk2-deviation of a skewed graph than on the dK-2 deviation of a less skewed graph. Note that the Youtube network has the largest skew, as the maximum degree is on the same order as the Flickr network, but the average degree is less.

References

1. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography. Commun. ACM **54**(12), 133–141 (2011)
2. Bollobás, B.: Random Graphs, Cambridge Studies in Advanced Mathematics, vol. 73, 2nd edn. Cambridge University Press, Cambridge (2001)
3. Caldarelli, G.: Scale-Free Networks: Complex Webs in Nature and Technology. Oxford University Press, Oxford (2013)
4. Chung, F., Lu, L.: Complex graphs and networks. In: CBMS Regional Conference Series in Mathematics, vol. 107. American Mathematical Society (2006)
5. Collberg, C.S., Kobourov, S.G., Carter, E., Thomborson, C.: Graph-based approaches to software watermarking. In: Bodlaender, H.L. (ed.) WG 2003. LNCS, vol. 2880, pp. 156–167. Springer, Heidelberg (2003)
6. Collberg, C., Thomborson, C.: Software watermarking: models and dynamic embeddings. In: ACM Symposium on Principles of Programming Language (POPL), pp. 311–324 (1999)

7. Eppstein, D., Goodrich, M.T., Lam, J., Mamano, N., Mitzenmacher, M., Torres, M.: Models and algorithms for graph watermarking. ArXiv ePrint abs/1605.09425 (2016). http://arxiv.org/abs/1605.09425
8. Hopper, N.J., Molnar, D., Wagner, D.: From weak to strong watermarking. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 362–382. Springer, Heidelberg (2007)
9. Khanna, S., Zane, F.: Watermarking maps: hiding information in structured data. In: 11th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 596–605 (2000). http://dl.acm.org/citation.cfm?id=338219.338612
10. Leskovec, J., Sosič, R.: SNAP: a general purpose network analysis and graph mining library in C++. http://snap.stanford.edu/snap
11. Mislove, A., Marcon, M., Gummadi, K.P., Druschel, P., Bhattacharjee, B.: Measurement and analysis of online social networks. In: 5th ACM/Usenix Internet Measurement Conference (IMC) (2007)
12. Mitzenmacher, M.: A brief history of generative models for power law and lognormal distributions. Internet Math. **1**(2), 226–251 (2004)
13. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. In: IEEE Symposium on Security and Privacy (SP), pp. 173–187 (2009)
14. Newman, M., Barabasi, A.L., Watts, D.J.: The Structure and Dynamics of Networks. Princeton Studies in Complexity. Princeton University Press, Princeton (2006)
15. Sala, A., Cao, L., Wilson, C., Zablit, R., Zheng, H., Zhao, B.Y.: Measurement-calibrated graph models for social network experiments. In: 19th International Conference on the World Wide Web (WWW), pp. 861–870 (2010)
16. Venkatesan, R., Vazirani, V.V., Sinha, S.: A graph theoretic approach to software watermarking. In: Moskowitz, I.S. (ed.) IH 2001. LNCS, vol. 2137, pp. 157–168. Springer, Heidelberg (2001)
17. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. CoRR abs/1205.6233 (2012). http://arxiv.org/abs/1205.6233
18. Zhao, X., Liu, Q., Zheng, H., Zhao, B.Y.: Towards graph watermarks. In: 2015 ACM Conference on Online Social Networks (COSN), pp. 101–112 (2015)
19. Zhao, X., Liu, Q., Zhou, L., Zheng, H., Zhao, B.Y.: Graph watermarks. ArXiv ePrint abs/1506.00022 (2015). http://arxiv.org/abs/1506.00022