K-Symmetry Model for Identity Anonymization in Social Networks

Wentao Wu Yanghua Xiao Wei Wang Zhenying He Zhihui Wang
School of Computer Science
Fudan University, Shanghai, China
{wentaowu, shawyh, weiwang1, zhenying, zhhwang}@fudan.edu.cn

ABSTRACT

With more and more social network data being released, protecting the sensitive information within social networks from leakage has become an important concern of publishers. Adversaries with some background structural knowledge about a target individual can easily re-identify him from the network, even if the identifiers have been replaced by randomized integers(i.e., the network is naively-anonymized). Since there exists numerous topological information that can be used to attack a victim's privacy, to resist such structural re-identification becomes a great challenge. Previous works only investigated a minority of such structural attacks, without considering protecting against re-identification under *any* potential structural knowledge about a target. To achieve this objective, in this paper we propose k-symmetry model, which modifies a naively-anonymized network so that for any vertex in the network, there exist at least $k-1$ structurally equivalent counterparts. We also propose sampling methods to extract approximate versions of the original network from the anonymized network so that statistical properties of the original network could be evaluated. Extensive experiments show that we can successfully recover a variety of such properties of the original network through aggregations on quite a small number of sample graphs.

1. INTRODUCTION

Social network, which consists of a set of entities representing individuals or organizations and relations among these entities, has been shown to be an invaluable tool to solve a variety of real applications including marketing, psychology and epidemiology. Recently, as more and more social network datasets published in one way or another, exploring the properties of these networks has attracted ever-increasing interests of researchers from different disciplines including sociology, physics, and computer science.

One of the fundamental issues when releasing social network data is avoiding disclosure of individuals' sensitive information while still permitting certain analysis on the network. A straightforward approach to achieve this objective is *naive anonymization*, which replaces all identifiers of individuals with randomized integers so that adversaries cannot directly locate each individual just according to his identifier. However, this simple strategy is insufficient[2, 4], since background knowledge of individuals such as degree [7], neighborhood graph [19], and so on, provides additional information which can be used by adversaries to re-identify the individuals from the naively-anonymized network.

All of those background knowledge mentioned above can be considered as *structural knowledge* of the corresponding individual, since each of them describes some information of the individual's topological connection to other individuals within the network. Adversaries having certain structural knowledge about an individual can re-identify him from the naively-anonymized network, provided that the candidate vertex matching the knowledge is unique. For instance, as shown in Figure 1, if we know that *Bob has 2 neighbors with degree 1*, then even all identifiers are removed, we can still identify Bob. Hay et.al [4] first formalize such identity disclosure based on structural knowledge of vertices as *Structural Re-identification*(SR).

To resist such identity disclosure, a reasonable solution is to modify the naively-anonymized network, denoted by G_a, so that there will be at least k entities satisfying the structural knowledge in the network after modification. This strategy is similar to the well known k-anonymity principle in traditional privacy-preservation technologies when releasing tabular data. Several previous works in this direction have proposed various k-anonymity models based on different structural knowledge used. For example, the k-degree anonymity model [7] modifies the network so that there are at least k vertices sharing the same degree, for each vertex; and the k-neighborhood anonymity model [19] modifies the network so that there are at least k vertices sharing isomorphic neighborhoods, for each vertex.

One fundamental problem underlying all of the models mentioned above is that each of them assumes some *specific* structural knowledge used by the adversaries in advance. However, in practice, it's very difficult for the network data publishers to make such predication since there exists numerous possible structural knowledge. On the other hand, as we shall see in Section 2.2, although descriptive power of certain structural knowledge may be limited, a combination of multiple easily obtained structural knowledge could have quite strong descriptive power, which can re-identify a large fraction of individuals from the network. So a k-anonymity model independent of structural knowledge used

EDBT 2010, March 22–26, 2010, Lausanne, Switzerland.

is necessary.

In this paper, we propose k-symmetry model to achieve this requirement. The general idea is to modify the network so that for each vertex v, there exist at least $k-1$ other vertices each of which serves as the image of v under some automorphism of the modified network. Informally speaking, an automorphism of a network is a *permutation* on its vertices which preserves its vertex adjacency relationships. In other words, the network remains invariant under the action of an automorphism. For instance, in Figure 1(b), if we exchange vertex 1 and 3 while fixing any other vertices, the vertex adjacency relationships of the network are conserved and therefore this permutation is an automorphism. Intuitively, any structural knowledge characterizing vertex 1 could also characterize vertex 3 and therefore they cannot be distinguished from each other by any structural knowledge. In Section 2.1, we shall formally demonstrate that such intuition really holds. We will then elaborate the technical details of the model, and further investigate several related problems, including anonymization algorithm, utility preservation, and possible improvements.

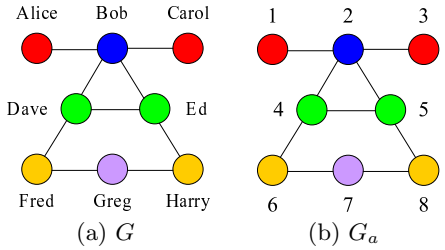
The rest of the paper is organized as follows: In Section 2, we present both theoretical analysis and experimental results to support the necessities of k-symmetry model, which have been shortly mentioned above. In Section 3, we formalize the k-symmetry model and then develop an anonymization procedure. The utility preservation problem is investigated in Section 4, where we propose two sampling approaches to extract approximate versions of the original network from the anonymized network. Section 5 further discusses several possible improvements on the basic k-symmetry model. Related works are summarized in Section 6, and we concludes the paper in Section 7.

2. MOTIVATION

In this section, we will first theoretically analyze the power of structural knowledge and illustrate why k-symmetry model could resist SR independent of the structural knowledge used. We then experimentally compare the power of single and combination of multiple structural knowledge. Both the theoretical analysis and the experimental results substantiate the necessities of k-symmetry model.

2.1 Power of Structural Knowledge

We first give some basic notations. A social network is modeled as a graph $G = (V(G), E(G))$, with $V(G)$ representing the set of entities and $E(G) \subseteq V(G) \times V(G)$ representing the set of edges, i.e. relations among the entities. For $v \in V(G)$, if $(u, v) \in E(G)$, then u is a *neighbor* of v, and we use $N(v)$ to denote the set of all neighbors of v. The cardinality of $N(v)$, i.e. $|N(v)|$, is the *degree* of v. Suppose π is a permutation on $V(G)$, for each $v \in V(G)$, we use v^π to denotes its image under π. Furthermore, we use $V(G)^\pi$ and $E(G)^\pi$ to denote the image of $V(G)$ and $E(G)$ under π, respectively. Clearly, $V(G)^\pi = V(G)$, and $E(G)^\pi = \{(u^\pi, v^\pi) | (u, v) \in E(G)\}$. An *automorphism* of graph $G(V(G), E(G))$ is a permutation π on $V(G)$ such that $G^\pi = G$, where $G^\pi = (V(G)^\pi, E(G)^\pi)$. Given two automorphisms f and g of G, the production of f and g is also an automorphism of G. Actually, the set of all automorphisms of G under the production of automorphisms forms a *group*, namely, the *automorphism group* of G, denoted by $Aut(G)$. Two vertices u, v of G are *automorphically equivalent* (denoted as $\sim$) to each other, if there exits an automorphism $g \in Aut(G)$ such that $u^g = v$. It's easy to verify that automorphism equivalence on vertices is an equivalence relation. The vertex partition induced by automorphism equivalence is called *automorphism partition* of G, denoted by $Orb(G)$, and each cell in $Orb(G)$ is called an *orbit* of $Aut(G)$. We use $Orb(v)$ to denote the orbit that vertex v belongs to.



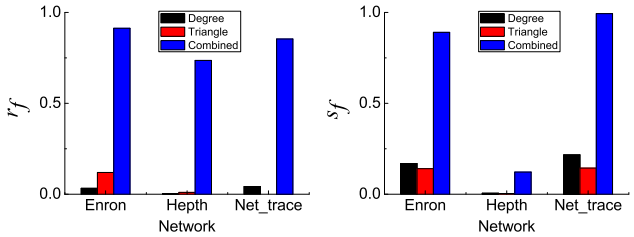
(a) G (b) G_a

Figure 1: Illustration of a social network G and its naively-anonymized version G_a.

Example 1 (Power of Structural Knowledge). *Figure 1 shows a network G and its naively-anonymized network G_a. Suppose the background structural knowledge is P_1:* Bob has at least 3 neighbors*, then the candidate set under P_1 is $\{2, 4, 5\}$. Thus, given the knowledge P_1, adversaries can identify Bob with probability 1/3. In contrast, if the structural knowledge about Bob is P_2:* Bob has 2 neighbors with degree 1*, then the candidate set under P_2 is $\{2\}$. Consequently, Bob can be uniquely re-identified from G_a by adversaries with P_2.*

After a social network G is naively-anonymized as G_a, any individual that a vertex $v \in V(G_a)$ represents (in the following texts, for notational convenience, we also use v to denote the corresponding individual) can be a *target* to be attacked if adversaries know some background structural knowledge about the individual. The knowledge could also be understood as some *assertion* of the individual, which could be evaluated to be *true* or *false* based on the topological structure of the network. Suppose P is one of such knowledge, by default, we set $P(v) = true$, then the power of P to re-identify vertex v can be accurately quantified by the size of the *candidate set of v under knowledge P*: $\mathbf{C}(P, v) = \{u | u \in V(G_a) \wedge P(u) = true\}$. Obviously, the smaller $|\mathbf{C}(P, v)|$ is, the easier that v could be re-identified from G_a, and consequently the more powerful P is for identifying v. In particular, the target v could be definitely re-identified under knowledge P if and only if $|\mathbf{C}(P, v)| = 1$. We illustrate the power of different structural knowledge describing the same individual in Example 1.

The key observation here is that, *for any structural knowledge P, $Orb(v) \subseteq \mathbf{C}(P, v)$*, since for any $u \in Orb(v)$, there exists an automorphism π such that $u^\pi = v$, which means $P(u) = P(u^\pi) = P(v) = true$ and therefore $u \in \mathbf{C}(P, v)$. Then, the power of any structural knowledge to re-identify a vertex from G_a is at most the size of the orbit to which the vertex belongs. In other words, *$|Orb(v)|$ is the upper bound for the power of any structural knowledge to re-identify vertex v from G_a.* For example, as shown in Figure 1(b), any two vertices in any one of the following vertex sets: $\{1, 3\}$, $\{4, 5\}$ and $\{6, 8\}$, cannot be distinguished from each other in G_a, no matter which structural knowledge is given, since $\{1, 3\}$, $\{4, 5\}$ and $\{6, 8\}$ are the orbits of the network.



(a) r_f (b) s_f

Figure 2: Ability of measures to re-identify a target.

Hence, if we modify the network G_a to be published so that for each vertex v, $|Orb(v)|$ is large enough, we can protect the privacy of identities against any possible SR. In other words, by modifying G_a to a graph G' such that for each vertex $v \in V(G')$, $|Orb(v)| \geq k$, we could achieve k-anonymity independent of structural knowledge used by adversaries. This is actually what k-symmetry model does, as we shall see in Section 3.

2.2 Combination of Structural Knowledge

Although constructing automorphic equivalence will guarantee the privacy of identities, we may still wonder whether this strategy is necessary for the privacy protection on real social networks. Next, we will show that by collecting multiple simple structural knowledge of a target, adversaries can obtain combined knowledge about the target and the size of the corresponding candidate set under such combined knowledge is quite close to that of the orbit to which the target belongs.

For example, we can define a combined structural measure as a two-tuple $f(v) = (Deg(v), tri(v))$, where $Deg(v)$ is the degree sequence (in the ascending or descending order) of vertices in the neighborhood of v, and $tri(v)$ is the number of triangles passing trough vertex v. Both of these two measures about a vertex can be easily obtained by an adversary. Note that any measure f on vertices implies an equivalence relation on $V(G)$, denoted as $\approx_f$, which is defined as $u \approx_f v (u, v \in V(G))$ *iff* $f(u) = f(v)$. Thus, we can further obtain a partition $\mathcal{V}_f$ induced by $\approx_f$.

To show the power of a measure f to re-identify a target from a naively-anonymized network, we calculate two statistics. One is

$$r_f = \frac{\sum_{V_i \in \mathcal{V}_f} \delta(V_i)}{\sum_{\Delta_i \in Orb(G)} \delta(\Delta_i)}$$

, where $\delta(V_i) = 1$ if $|V_i| = 1$ and $\delta(V_i) = 0$ otherwise. r_f quantifies the relative power of f to uniquely re-identify a target from G_a. We also define

$$s_f = \frac{\sum_{\Delta_i \in Orb(G)} |\Delta_i|(|\Delta_i| - 1)}{\sum_{V_i \in \mathcal{V}_f} |V_i|(|V_i| - 1)}$$

, which is the similarity between $\mathcal{V}_f$ and $Orb(G)$. If s_f is close to 1, the power of f to re-identify a target is close to the upper bound of any structural knowledge.

For comparison, we also summarize values of s_f and r_f for $deg(v)$(degree of vertex v) and $tri(v)$ on three real social networks: **Enron**, **Hepth** and **Net_trace**. The results are shown in Figure 2, from which we can clearly see that the re-identification power of the combined measure, either in the average sense (quantified by s_f) or in the strongest sense(quantified by r_f), is quite close to the upper bound. Hence, in practice, it is necessary to anonymize a network by constructing automorphic equivalence relation on G_a, which motivates us to propose the k-symmetry model.

3. K-SYMMETRY MODEL

In this section, we will first formalize k-symmetry model in 3.1, then define orbit copying operation to implement k-symmetry in Section 3.2. The detailed anonymization procedure is described in Section 3.3.

3.1 Problem Definition

Definition 1 (k-Symmetry Anonymity). *Given a graph G and an integer k, if $\forall \Delta \in Orb(G)$, $|\Delta| \geq k$, then G is **k-symmetric**, or, G satisfies the requirement of **k-symmetry anonymity**.*

K-symmetry anonymity is a generalization of any other k-anonymities of graphs based on different structural constraints on vertices. In other words, if a graph is k-symmetric, it also satisfies any other k-anonymity requirements defined in terms of other structural constraints on vertices, such as degree, neighborhoods and so on.

Then the problem becomes: *Given a graph G and an integer k, how to modify G so that the resulting graph G' is k-symmetric?* In this paper, we will only consider vertex/edge insertion as the graph modification operations. Consequently, the original graph G must be a subgraph of the anonymized graph G'.

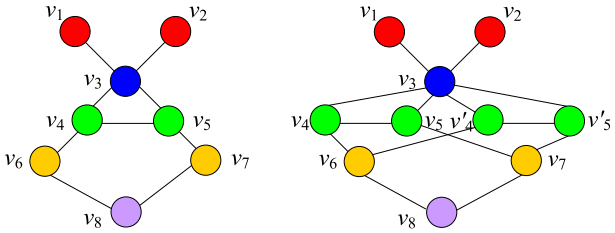
3.2 Orbit Copying Operation

Since the vertices in each orbit are already automorphically equivalent to each other, our basic idea to modify a graph G to be *k-symmetric* is then to make duplicate copies of each orbit in $Orb(G)$, until the total size of each orbit combined with its copies is at least k. However, such copying is not trivial, since we need to ensure that all the vertices in the union now are still automorphically equivalent. In this section, we will formalize the definition of our *orbit copying* operation, and show that the above requirement is satisfied.

We first need to generalize the concept of automorphism partition to *sub-automorphism partition*, which underlies the definition of orbit copying operation as well as the following theoretic analysis.

Definition 2 (Sub-automorphism partition). *Let G be a graph and $\mathcal{V}$ be a vertex partition on $V(G)$. $\mathcal{V}$ is a **sub-automorphism partition** of G if $\forall O \in \mathcal{V}$, $\forall u, v \in O$, $\exists g \in Aut(G)$ such that $u^g = v$ and $\mathcal{V}^g = \mathcal{V}$.*

Clearly, if $\mathcal{V}$ is a sub-automorphism partition of G, then $\mathcal{V}$ is finer than $Orb(G)$, which means that for each $V_i \in \mathcal{V}$, there must exist some $\Delta_j \in Orb(G)$ such that $V_i \subseteq \Delta_j$. In particular, $Orb(G)$ is also a sub-automorphism partition of G. Hence, sub-automorphism partition can be considered as a generalization of automorphism partition. It's worthwhile to point out that such generalization is not trivial since we can find many partitions finer than automorphism partition that cannot be classified as sub-automorphism partitions, which is illustrated in Example 2.



(a) Graph G (b) Graph G' after V_3 is copied

Figure 3: Illustration of orbit copying operation.

EXAMPLE 2 (SUB-AUTOMORPHISM PARTITION). *Consider a cyclic graph with four vertices $\{1,2,3,4\}$ and edge set $\{(1,2)(2,3)(3,4)(1,4)\}$. It's easy to check that the partition $\{\{1,2\},\{3,4\}\}$ is a sub-automorphism partition. But the partition $\{\{1,2,3\},\{4\}\}$ is not, since we cannot find any automorphism that maps 2 to 3 while fixing the partition.*

Now, we give the formal definition of orbit copying operation (in Definition 3), which is illustrated in Example 3.

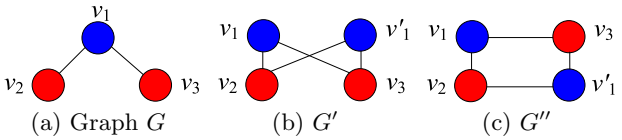
DEFINITION 3 (ORBIT COPYING). *Given a graph G and a sub-automorphism partition $\mathcal{V}$ of G. Suppose $V \in \mathcal{V}$, an **orbit copying** operation $Ocp(G,\mathcal{V},V)$ is defined as follows:*
For each $v \in V$, introduce a new vertex v' into graph G and:
1.if $(u,v) \in E(G)$, $u \in U$, $U \in \mathcal{V}$ and $U \neq V$, then add an edge (u,v') into G;
2.if $(u,v) \in E(G)$, $u \in V$, then add an edge (u',v') into G.

EXAMPLE 3 (ORBIT COPYING). *As shown in Fig 3(a), the original graph G has the automorphism partition $Orb(G) = \{V_1,V_2,V_3,V_4,V_5\}$, where $V_1 = \{v_1,v_2\}$, $V_2 = \{v_3\}$, $V_3 = \{v_4,v_5\}$, $V_4 = \{v_6,v_7\}$ and $V_5 = \{v_8\}$. Fig 3(b) shows the graph after the orbit V_3 is copied.*

Note that usually we will use $Orb(G)$ as the initial partition before any orbit copying operation. The key point of the orbit copying operation is that the copy of the original orbit can strictly preserve the adjacency relation between the original orbit and other orbits. For instance, as shown in Example 3, V'_3 is still adjacent to V_2 and V_4. Hence, in the graph G' after an orbit copying operation $Ocp(G,\mathcal{V},V)$, any vertex in V is automorphically equivalent to its copy in V'. As a result, in G', all vertices in $V \cup V'$ will be automorphically equivalent to each other. Therefore, we must have $V \cup V' \subseteq \Delta_i$, for some $\Delta_i \in Orb(G')$. We also need to notice that $V \cup V'$ is not necessarily an orbit of $Orb(G')$, which is illustrated in Example 4.

EXAMPLE 4 ($\mathcal{V}'$ AND $Orb(G')$). *As shown in Figure 3, after orbit copying, we have $\mathcal{V}' = Orb(G')$. However, Figure 4 gives a counterexample. Here, $\mathcal{V} = Orb(G) = \{V_1,V_2\}$ with $V_1 = \{v_1\}$ and $V_2 = \{v_2,v_3\}$. After V_1 is copied, we obtain G' and $\mathcal{V}' = \{\{v_1,v'_1\},\{v_2,v_3\}\}$. If G' is laid out as G'', it's easy to check that all the four vertices of G' belong to the same orbit of G'. Therefore, $\mathcal{V}' \neq Orb(G')$.*

We formalize the above intuitions about orbit coping operation in Lemma 1. Due to space limitations, all the proofs are omitted in this paper.



(a) Graph G (b) G' (c) G''

Figure 4: An example that $\mathcal{V}' \neq Orb(G')$.

LEMMA 1. *Let G be a graph and $\mathcal{V} = \{V_1,V_2,...,V_m\}$ be a sub-automorphism partition of G. After an orbit copying operation $Ocp(G,\mathcal{V},V_i)(1 \leq i \leq m)$, the partition $\mathcal{V}^{(1)} = \{V_1,V_2,...,V_i^{(1)},...,V_m\}$ is a sub-automorphism partition of $G^{(1)}$, where $V_i^{(1)}$ is the union of V_i and its copy, and $G^{(1)}$ is the resulting graph.*

Since our ultimate goal is to modify a network to be k-symmetric, only one copying operation does not necessarily ensure the size of the augmented cell is large enough. Hence, we may need to perform orbit copying operations on the same cell of the initial partition multiple times to achieve the size requirement. An immediate consequence of Lemma 1, described in Lemma 2, guarantees that the resulting partition by merging all copies as well as the copied cell is still a sub-automorphism partition of the resulting graph.

LEMMA 2. *Let G be a graph and $\mathcal{V} = \{V_1,V_2,...,V_m\}$ be a sub-automorphism partition of G. The vertex partition after applying $N \geq 0$ orbit copying operations $Ocp(G,\mathcal{V},V_i)$ on the same cell $V_i(1 \leq i \leq m)$, i.e., $\mathcal{V}^{(N)} = \{V_1,V_2,...,V_i^{(N)},...,V_m\}$ is a sub-automorphism partition of the resulting graph $G^{(N)}$, where $V_i^{(N)}$ is the union of V_i and all its copies (in particular, $V_i^{(0)} = V_i$) and $G^{(N)}$ is the resulting graph.*

Lemma 1 and 2 only focus on the orbit copying operations on a single cell, but to achieve the k-symmetry anonymity, we may have to perform a series of orbit copying operations on different cells in the initial partition. Let $\mathbf{O} = O_1...O_N$ be any orbit copying operation sequence of length N performed on G, where $O_n = Ocp(G,\mathcal{V},V_{i_n})$ for $1 \leq n \leq N$, and let the graph produced by $\mathbf{O}$ be $G_{\mathbf{O}}$. Suppose π is a permutation on the set $\{1,2,...,N\}$. Let the operation sequence under π be $\pi(\mathbf{O}) = O_{\pi(1)}...O_{\pi(N)}$, where $O_{\pi(n)} = Ocp_{\pi(n)}(G,\mathcal{V},V_{i_{\pi(n)}})$, for $1 \leq n \leq N$. The next lemma shows that orbit copying operation is order-independent.

LEMMA 3. *Let G be a graph, and $\mathcal{V} = \{V_1,V_2,...,V_m\}$ be a sub-automorphism partition of G. Suppose $\mathbf{O}$ is any orbit copying operation sequence of length N performed on G. Let α and β be any two permutations on the set $\{1,2,...,N\}$, then $G_{\alpha(\mathbf{O})}$ and $G_{\beta(\mathbf{O})}$ are* isomorphic.

Now we are ready to show an important theorem which states that an arbitrary sequence of orbit copying operations on the initial partition will always produce a sub-automorphism partition of the resulting graph. It is a generalization of Lemma 2, and will be the foundation of our anonymization procedure proposed in the next subsection.

THEOREM 1. *Let G be a graph and $\mathcal{V} = \{V_1,V_2,...,V_m\}$ be a sub-automorphism partition of G. Suppose $\mathbf{O}$ is any orbit copying operation sequence of length N performed on G. Let the resulting vertex partition and the corresponding graph be $\mathcal{V}^{(N)}$ and $G^{(N)}$, where each cell in $\mathcal{V}^{(N)}$ is the union of the original orbit and all of its copies. Then $\mathcal{V}^{(N)}$ is a sub-automorphism partition of $G^{(N)}$.*

3.3 Anonymization Procedure

Based on the orbit copying operations, we now propose an anonymization procedure to modify a graph to be k-symmetric, which is shown in Algorithm 1. The basic idea of the anonymization is repeating the orbit copying operation for each $V_i \in Orb(G)(|V_i| \leq k)$ until the size of the union of V_i and its copies are equal to or larger than k.

Algorithm 1: Anonymization

Input: a graph G and its automorphism partition $Orb(G) = \{V_1, V_2, ..., V_m\}$; the specified threshold k
Output: a k-symmetric graph G' with respect to G and $Orb(G)$

```
for 1 ≤ i ≤ m do
    if |V_i| ≥ k then
        Continue;
    end
    else
        Let V_i' = V_i;
        while |V_i'| < k do
            Ocp(G, Orb(G), V_i);
            V_i' = V_i' ∪ V_i;
        end
    end
end
```

The fact that the graph produced by Algorithm 1 is k-symmetric is a straightforward result of Theorem 1, which is formally claimed in Theorem 2.

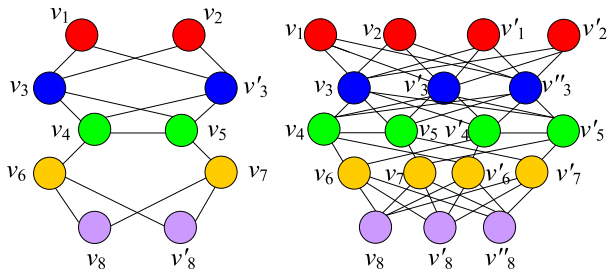
THEOREM 2. *The graph G' produced by the anonymization procedure is k-symmetric.*

We illustrate the anonymization procedure as well as its major properties in Example 5.

EXAMPLE 5 (ANONYMIZATION PROCEDURE). *Consider the graph G shown in Figure 3(a). Suppose now $k = 2$, then V_2 and V_5 need to be copied if we want to produce a 2-symmetric graph. Figure 5(a) shows the graph G' after the anonymization procedure. Now we obtain a new vertex partition $\mathcal{V}' = \{V_1, V_2, V_3, V_4, V_5\}$ of $V(G')$, where $V_2 = \{v_3, v'_3\}, V_5 = \{v_8, v'_8\}$ and other cells remain unchanged. Each cell of $\mathcal{V}'$ contains at least 2 vertices that are structurally equivalent and we can easily check that $\mathcal{V}'$ is a sub-automorphism partition of G'. Figure 5(b) shows the graph G' after the anonymization procedure when $k = 3$. Here, since none of the 5 orbits of $Orb(G)$ satisfies the k-symmetry constraint, all of them need to be copied.*

In general, the order of orbits in $Orb(G)$ is not necessarily to be unique. However, attributed to Lemma 3, our anonymization procedure is independent of the orbit copying order, which means that we could always obtain the same k-symmetric graph G' whatever the order of orbits in $Orb(G)$ is.

The time complexity of the anonymization procedure is polynomial. Note that the time complexity of Algorithm 1 relies on the number of vertices and edges newly added into the network. Specifically, suppose there are N orbits in $\mathcal{V}$ containing less than k vertices, denoted by $V_{i_1}, ..., V_{i_N}$. Let $k_1, ..., k_N$ be the number of orbit copying operations for each orbit $V_{i_k} (1 \leq k \leq N)$, respectively. Then the total number of vertices added is $\sum_{j=1}^{N} k_j |V_{i_j}| \leq (k-1)|V(G)|$, since each $k_j \leq k-1$ and $\sum_{j=1}^{N} |V_{i_j}| \leq |V(G)|$. And the total number of edges introduced is less than $\sum_{j=1}^{N} k_j |V_{i_j}|(k|V(G)|) \leq k(k-1)|V(G)|^2$. Note that usually k is much smaller than $|V(G)|$ and thus could be treated as a constant. Hence, the worst case running time of the anonymization procedure is $O(|V(G)|^2)$.



(a) G' when $k = 2$ (b) G' when $k = 3$

Figure 5: Illustration of anonymization procedure.

4. UTILITY

A critical problem that needs to be addressed in any privacy protection model is the *utility* of the published data. One of the desired utilities is permitting users summarizing key statistical properties of the original network on the published network data. We will publish a graph that is anonymized to be k-symmetric so that adversaries cannot re-identify any vertex. We further provide graph-backbone (discussed in Section 4.1) based sampling approaches to extract approximate versions of the original network from the anonymized network, so that analysts can evaluate approximate values of the key properties of the original network from these sample graphs.

4.1 Graph Backbone

One of the very recent progress in network science shows that the skeleton by collapsing all automorphically equivalent classes in the network can preserve many key properties of original network including diameter, average shortest path length and hub vertices [15]. Hence, preserving such skeleton when anonymizing a network will be crucial for analysts to recover certain key statistical properties of the original network. Note that definition of orbit copying implies that the linkage pattern between orbits in the original network can be precisely preserved. Thus, it's reasonable to expect that the original network and the anonymized network will share the same or similar skeleton in the sense of filtering out the structurally equivalent vertices in the network. And consequently, sampling approach on the anonymized k-symmetric network under certain heuristics can help us capture the skeleton of the original network, thus providing the opportunity for users to accurately recover the statistical information of the original network.

However, [15] only shows a general framework to capture the graph skeleton by coarse-graining all orbits of a graph without considering many special cases where the structure will collapse after reduction of certain orbits. We will show many cases where it's necessary to exert restriction on the reduction so that we can get a more meaningful skeleton, which is consequently expected to preserve more structural properties of the original network. Motivated by these intu-

itions, in this section we will first propose a new structural skeleton of a network, called as *graph backbone*, which is closely related to the orbit copying operation.

Graphs under orbit copying operation implies certain relation on graphs. Let $(H, \mathcal{V}_H)$ be a two-tuple with H representing a graph and $\mathcal{V}_H$ be a sub-automorphism partition of H. Thus, after the action of a set of orbit copying operations on cells within $\mathcal{V}_H$, we can obtain a unique two-tuple $(G, \mathcal{V}_G)$ with G representing the resulting graph and $\mathcal{V}_G$ representing the resulting partition of G. Then, we call $(G, \mathcal{V}_G)$ as a *generalization* of $(H, \mathcal{V}_H)$, and $(H, \mathcal{V}_H)$ as a *reduction* of $(G, \mathcal{V}_G)$. Such relation is denoted by $(H, \mathcal{V}_H) \leq (G, \mathcal{V}_G)$. In the context without confusion, we also say H is a reduction of G, or G is a generalization of H, without explicitly specifying corresponding partitions of G and H. In particular, G is both a generalization and reduction of itself. Obviously, the resulting graph G' of the anonymization procedure is a generalization of the input graph G, with respect to $Orb(G)$.

First, it's clear that a graph G may have multiple reductions, with respect to the given sub-automorphism partition $\mathcal{V}_G$. Let $\mathcal{R}(G, \mathcal{V}_G)$ be the set consisting of all the reductions of $(G, \mathcal{V}_G)$, namely, $\mathcal{R}(G, \mathcal{V}_G) = \{(H, \mathcal{V}_H) | (H, \mathcal{V}_H) \leq (G, \mathcal{V}_G)\}$. Since $(G, \mathcal{V}_G) \in \mathcal{R}(G, \mathcal{V}_G)$, $\mathcal{R}(G, \mathcal{V}_G)$ is not empty. Clearly, the reduction relation $\leq$ on $\mathcal{R}(G, \mathcal{V}_G)$ is reflexive, asymmetric, and transitive. Hence $\leq$ is a partial order on $\mathcal{R}(G, \mathcal{V}_G)$. Then, we are ready to give Theorem 3, which is a fundamental property of the poset $(\mathcal{R}(G, \mathcal{V}_G); \leq)$.

THEOREM 3. *The poset $(\mathcal{R}(G, \mathcal{V}_G); \leq)$ is a bounded lattice.*

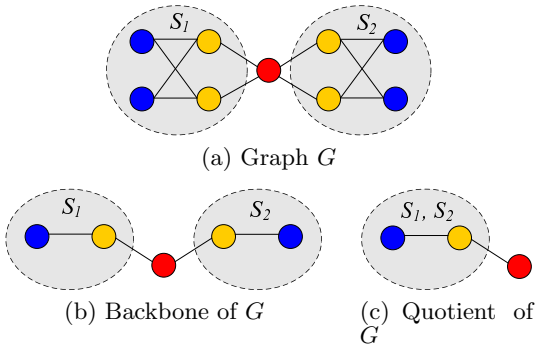
The graph backbone is then defined as the least element in $(\mathcal{R}(G, \mathcal{V}_G); \leq)$, which is formally shown in Definition 4.

DEFINITION 4 (GRAPH BACKBONE). *Given a graph G and a sub-automorphism partition $\mathcal{V}_G$ of G. The **backbone** of $(G, \mathcal{V}_G)$ is the least element in the bounded lattice $(\mathcal{R}(G, \mathcal{V}_G); \leq)$.*

Since $(\mathcal{R}(G, \mathcal{V}_G); \leq)$ is a bounded lattice, the uniqueness of graph backbone of $(G, \mathcal{V}_G)$ can be ensured. Usually, we denote the unique backbone of $(G, \mathcal{V}_G)$ as $B_{G,\mathcal{V}_G}$.

EXAMPLE 6 (GRAPH BACKBONE). *Figure 6 illustrates a graph G and its backbone $B_{G,\mathcal{V}_G}$. Note that vertices in G are colored in terms of corresponding sub-automorphism partition $\mathcal{V}_G$.*

Here, we need to give some remarks on the definition of graph backbone. The backbone of $(G, \mathcal{V}_G)$ is essentially the smallest skeleton that can be used as the seed graph from which the network can grow to be $(G, \mathcal{V}_G)$ through orbit copying operations. Hence, if we define an inverse operation of orbit copying operation, the backbone of $(G, \mathcal{V}_G)$ is just the ultimate result after a sequence of such inverse operations. For the convenience of description, we call such inverse operation as *reduction* operation. Clearly, the reduction operation sequence from $(G, \mathcal{V}_G)$ to its backbone is not unique. Such a reduction operation simply coarse-grain an automorphic equivalence class. However, each orbit copying operation or its inverse operation involves only one orbit in each operation. Thus, two automorphic substructures involving more than one orbits cannot be reduced anymore in a backbone. In contrast, in the network quotient reduction [15], there exists no such restriction. For example, as shown in Figure 6, the two isomorphic subgraphs S_1 and S_2 of G will be preserved in G's backbone, however, they will be reduced to one in G's quotient.



(a) Graph G

(b) Backbone of G

(c) Quotient of G

Figure 6: Illustration of graph backbone and quotient.

Clearly, such restriction in the backbone reduction is more meaningful, since usually different isomorphic substructures are deemed as different modules of the network. In the previous example, S_1 and S_2 are two obvious modules of G. Hence, when seeking G's reduction, it's more reasonable to preserve such modular information.

Now we can move to another property of backbone: *the orbit copying operation can preserve the backbone*, which means that after the action of any sequence of orbit copying operations, the resulting graph can be reduced to the same backbone, which is formally stated in Theorem 4. Thus, the graph G' produced by the anonymization procedure will preserve the backbone of the original graph G, with respect to the given sub-automorphism partition $\mathcal{V}$ of G.

THEOREM 4. *Let $(H, \mathcal{V}_H) \leq (G, \mathcal{V}_G)$, i.e., $(G, \mathcal{V}_G)$ be a generalization of $(H, \mathcal{V}_H)$. Then $B_{G,\mathcal{V}_G} = B_{H,\mathcal{V}_H}$.*

4.2 Backbone-Based Sampling

Since backbone captures the essential structural properties of the original network, graphs with the same backbone and similar size tend to have the same or similar statistical properties. Thus, the key to approximately recover the structure of the original network is to extract its backbone from its k-symmetric version, which is possible since graph G and its k-symmetric anonymized graph share the same backbone(Theorem 4). In this section, we will propose two backbone-based sampling strategies to extract the original network from the anonymized network. We first outline the general framework of backbone-based sampling in 4.2.1, then propose the exact backbone-based sampling and approximate backbone-based sampling in 4.2.2 and 4.2.3, respectively.

4.2.1 A General Framework

Let $(G', \mathcal{V}')$ be the result after anonymization of $(G, \mathcal{V}_G)$. Suppose $\mathcal{P}$ is a set of knowledge of G, for example the number of vertices or edges of G. Similarly as in Section 2.1, here we can model each knowledge $P \in \mathcal{P}$ as some *assertion* of G. By default, we set $P(G) = true$. Then the set $SS(G', \mathcal{V}', \mathcal{P}) = \{(H, \mathcal{V}_H) | B_{G',\mathcal{V}'} = B_{H,\mathcal{V}_H} \wedge (P(H) = true, \forall P \in \mathcal{P})\}$ forms the *sample space*, with respect to $\mathcal{P}$. The backbone-based sampling strategy simply takes a graph uniformly from the sample space.

Clearly, using different knowledge $\mathcal{P}$ will result in different sample spaces. Since the original graph G is also in the sample space, sample spaces with large size are preferred. But on the other hand, larger sample space usually means less restrictions on the structure of the sampled graph, which may degrade its utility. Therefore, it is left to the network publisher to choose a reasonable set $\mathcal{P}$ that achieves good trade-off between sample space size and sample utility. However, simply using the knowledge of $|V(G)|$ has already provided such a trade-off. In this case, suppose the corresponding sub-automorphism partition of $B_{G',\mathcal{V}'}$ is $\mathcal{B} = \{B_1, B_2, ..., B_m\}$. The size of the sample space then equals to the number of feasible solutions $(k_1, ..., k_m)$ to the equation $\sum_{i=1}^{m} k_i|B_i| = |V(G)|$, where each k_i is a positive integer. Thus, in general the size of sample space is expected to be exponential to $|V(G)|$. As a result, even in a network with moderate size, the sample space size will be unimaginably large. However, as we shall see later in the experimental section, the sample utility is surprisingly good in most cases. Hence, in the following implementations of the backbone-based sampling strategy, we only consider the cases where knowledge about the number of vertices in the original network will be published.

4.2.2 Sampling Based on Exact Backbone

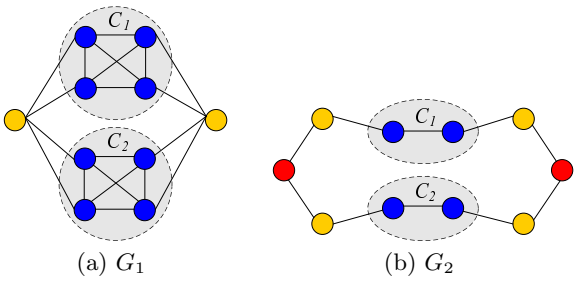
We next propose the exact backbone detection algorithm, which is shown in Algorithm 2. The basic idea of the algorithm is to reduce the network by repeatedly removing the subgraphs which is obtained by orbit copying operations (line 8 to 12). However, we first need to identify subgraphs constructed by orbit copying operation (line 2 to 7).

Algorithm 2: Graph Backbone Detection

```
   Input: G, V
   Output: B_{G,V}
1  foreach V ∈ V do
2      foreach v ∈ V do
3          Compute L(v);
4          foreach u ∈ L(v) do
5              Add (v, u) into L(V);
6          end
7      end
8      foreach C ∈ C(G_V) do
9          if ∃C' ∈ C(G_V), G'_C ≅_{L(V)} G_C then
10             Remove C' from G;
11         end
12     end
13 end
14 return The resulting graph, which must be B_{G,V};
```

Let G be a graph and $\mathcal{V}$ be one of its sub-automorphism partitions. From the definition of orbit copying operation, we have that the subgraph induced by each cell $V \in \mathcal{V}$, denoted as $G[V]$, will consist of a set of connected components and some of the components can be considered as copies of the remaining components. Let $\mathcal{C}(G[V])$ be the set of $G[V]$'s components. If one component $C_1 \in \mathcal{C}(G[V])$ is the copy of $C_2 \in \mathcal{C}(G[V])$, C_1 must be isomorphic to C_2. For example, as shown in Figure 7, in both of the two graphs, the component C_1 is isomorphic to the component C_2 (C_1 and C_2 are the induced subgraphs of the cell whose vertices are marked by blue).

However, in some cases, isomorphic relation between components is not sufficient to characterize the orbit-copying



(a) G_1 (b) G_2

Figure 7: Examples of graphs and their backbones.

relation between components. For example, in Figure 7(b), component C_1 (C_2) will not be an orbit-copy of C_2 (C_1), since no vertex in C_1 shares the same neighbor with any vertex in C_2. However, in Figure 7(a), for each vertex $u \in C_1$, we can find a corresponding vertex $v \in C_2$ such that they share the same neighbors that does not lie in the blue cell. Let $\mathcal{L}(V)$ be the set of all vertex pairs (u, v) such that they share the same neighbors outside V, that is $N(v) \cap (V(G) - V) = N(u) \cap (V(G) - V)$, and they come from different components of $\mathcal{C}(G[V])$. Then, only if we can find an isomorphism θ from $C_i \in \mathcal{C}(G[V])$ to $C_j \in \mathcal{C}(G[V])$ such that each $(u, \theta(u)) \in \mathcal{L}(V)$ (such a relation between C_1 and C_2 is denoted by $C_1 \cong_{\mathcal{L}(V)} C_2$), C_i (C_j) will be an orbit-copy of C_j (C_i) and thus can be removed (line 9 to 11). When we remove a component from the network, we remove the vertices contained in the component as well as all their incident edges.

Above analysis shows that the components removed are strictly the orbit copies of some components in the same cell, thus we can ensure that the resulting graph is the backbone of $(G, \mathcal{V})$. Based on the above identification algorithm of graph backbone, we can now further propose a sampling strategy to approximate the original network, which is illustrated in Algorithm 3.

Algorithm 3: Exact backbone-based sampling

```
   Input: G',V', n = |V(G)|, p[1...|V'|]
   Output: A connected subgraph G_s of G' such that
           |V(G_s)| ≈ n
1  Compute B_{G',V'};
2  N = n − |V(B_{G',V'})|;
3  while N > 0 do
4      Randomly pick i with probability p[i] such that
       (CPN[i] + 1) · |B_i| ≤ |V'_i|, where 1 ≤ i ≤ |V'|, B_i ∈ B
       and V'_i ∈ V';
5      CPN[i] = CPN[i] + 1;
6      N = N − |B_i|;
7  end
8  for 1 ≤ i ≤ |B| do
9      Repeat Ocp(B_{G',V'}, B, B_i) CPN[i] times;
10 end
11 return The resulting graph as G_s;
```

The input of Algorithm 3 is the generalization $(G', \mathcal{V}')$ of the original network G and the number of vertices of G. Here, users can also specify $p[i]$ as the input, which is the *sampling probability* from cell V'_i of $\mathcal{V}'$. In general, $p[i]$ can follow any distribution. However, real social networks usually have a right-skewed degree distribution, meaning that the number of vertex with degree k is inversely correlated to

k. Thus, in the original partition $\mathcal{V}$, the size of cells is expected to be inversely related to the degree of vertices in the cell. Hence, usually, users can define $p[i] = d_i^{-1} / \sum_{j=1}^{|\mathcal{V}'|} d_j^{-1}$, where d_i is the degree of vertices in V_i'.

The basic idea of the algorithm is to distribute $N = n - |V(B_{G',\mathcal{V}'})|$ vertices into different cells of $\mathcal{B}$ with probability $p[i]$. In the real implementation, we first compute $CPN[i]$ (initialized as 0) to record the number of orbit copying operations that should be performed on cell B_i of $\mathcal{B}$ according to $p[i]$ (line 3 o 7). Then, we repeat the orbit copying operation $CPN[i]$ times for each cell in $\mathcal{B}$.

Note that the number of vertices in the resulting graph may be slightly more than $|V(G)|$, but the number of additional vertices inserted will not exceed the size of the cell chosen at the last iteration of the while loop. Introduction of extra vertices usually can be ignored since most cells in the automorphism partition of a real network are of very small size compared to the whole population.

The major weakness of this implementation is its potential inefficiency. Note that when calculating graph backbone in Algorithm 2, indeed we need to perform graph isomorphism testing, whose complexity is still an open problem. Specifically, neither we have found a polynomial time algorithm, nor we can prove that the problem is NP-complete [3]. Therefore, in the worst case, it is unlikely that there exists an efficient algorithm outperforming the brute-force search.

4.2.3 Sampling Based on Approximate Backbone

To reduce computational complexity, in this subsection, we propose an alternative implementation with linear time complexity in the worst case.

The procedure is illustrated in Algorithm 4, whose inputs are the same as that of Algorithm 3. In the procedure, $S[i]$(initialized as 1) records the number of vertices that should be sampled from cell $V_i \in \mathcal{V}'$, according to $p[i]$; $Visited[i]$ and $Selected[i]$ are two booleans, respectively, indicating whether vertex v_i is visited and selected in the DFS procedure shown in Algorithm 5; all other notations have the same meaning as they are in Algorithm 3.

Algorithm 4: Approximate backbone-based sampling

Input: G', $\mathcal{V}'$, $n = |V(G)|$, $p[1...|\mathcal{V}'|]$
Output: A connected subgraph G_s of G' such that $|V(G_s)| = n$

```
1  N = n − |V'|;
2  while N > 0 do
3  |  Randomly pick i with respect to p[i] such that
   |  S[i] < |V_i'|, where 1 ≤ i ≤ |V'| and V_i' ∈ V';
4  |  S[i] = S[i] + 1;
5  |  N = N − 1;
6  end
7  Uniformly pick a vertex r ∈ V(G'), and suppose r in V_j';
8  Visited[r] = true;
9  Selected[r] = true;
10 S[j] = S[j] − 1;
11 n = n − 1;
12 DFS(r, n, Visited, Selected, S, V');
13 return The subgraph induced by
   V = {v|v ∈ V(G') ∧ Selected[v] = true});
```

The main idea of the sampling procedure is to sample overall n vertices from cells of $\mathcal{V}'$ through a depth-first traversal on the graph G' and return the subgraph induced by such n sampled vertices as the approximation of the original network. For each cell in $\mathcal{V}'$, we first need to compute the expected number of sampled vertices, which is proportional to the given probability $p[i]$. Then, we randomly select a vertex in G' as the root of the resulting DFS-tree (line 7 to 11 in Algorithm 4), then use $S[1, ..., |\mathcal{V}'|]$ to guide the DFS procedure, such that for each cell V_i at most $S[i]$ vertices will be sampled (from line 8 to 13 in Algorithm 5). The rationality of exploiting the DFS as the framework of vertex sampling is to ensure the connectedness of the induced subgraph, which is required by the original graph.

Algorithm 5: $DFS(v, n, Visited, Selected, S, \mathcal{V}')$

Input: v, n, $Visited$, $Selected$, S, $\mathcal{V}'$

```
1  foreach u ∈ N(v) do
2  |  if n < 1 then
3  |  |  return;
4  |  end
5  |  if !Visited[u] then
6  |  |  Visited[u] = true;
7  |  |  //Suppose u ∈ V_t'.
8  |  |  if S(t) > 0 then
9  |  |  |  Selected[u] = true;
10 |  |  |  S[t] = S[t] − 1;
11 |  |  |  n = n − 1;
12 |  |  |  DFS(u, n, Visited, Selected, S, V');
13 |  |  end
14 |  end
15 end
```

This implementation tries but cannot guarantee to fully capture the backbone of G'. As a result, the sample graph produced is not assured to come from the sample space $SS(G', \mathcal{V}', \mathcal{P})$. However, as shown in the following experimental section, sampling based on heuristic DFS can closely approximate the structure of the original network. Another obvious advantage of this implementation is its efficiency. Since it is in fact a depth-first traversal of G' plus some preprocessing, the worst case running time is $O(|V(G')| + |E(G')|)$, which is linear.

4.3 Experiments of Backbone-based Sampling

In this section, we provide extensive experiments to show the effect of our backbone-based sampling approach. Three real network datasets, **Hepth**, **Enron** and **Net_trace**, are used in our experiments, which are also used in [4]. Table 1 lists the basic statistics of these real networks. For our purpose of backbone-based sampling, we release the anonymized network G', the corresponding sub-automorphism partition $\mathcal{V}'$ as well as $|V(G)|$ to the public.

We only focus on the utility of the statistical properties of the original graph. Specifically, similar to [4], we consider an analyst who estimates a graph property by drawing sample graphs from G', measuring the property of each sample, and then aggregating measurements across samples. We examine four properties commonly measured and reported on network data. *Degree* is the degree distribution of the graph. *Path length* is a distribution of the lengths of the shortest paths between 500 randomly sampled pairs of vertices in the network. *Transitivity* (or, *clustering coefficient*) is a distribution of clustering coefficients of all vertices. Clustering coefficient of a vertex is defined as the proportion of connected neighbor pairs among all possible neighbor pairs. *Network resilience* is measured by plotting the fraction of the number of vertices contained in the largest connected component as

Table 1: Statistics of networks used.

Statistic	Network		
	Hep-Th	Enron	Net-trace
Number of vertices	2510	111	4213
Number of edges	4737	287	5507
Minimum degree	1	1	1
Maximum degree	36	20	1656
Median degree	2	5	1
Average degree	3.77	5.17	2.61

vertices are removed in descending order of degree [1].

We measured each of these characteristics for the original graph G and for a set of 20 output graphs produced by both Algorithm 3 and 4 with $k = 5$. We are surprised to find that the results produced by the two strategies are almost the same. What's more, the approximation algorithm (Algorithm 4) performs even a bit better than Algorithm 3 in the case of **Hepth** and **Net_trace**. Such observations can be naturally interpreted since in Algorithm 3, when a vertex sampled has large degree, a large number of edges should then be copied, which will significantly degrade the approximation effect if the vertex selected actually does not have automorphically equivalent counterparts in the original network. Due to the similarity of the results produced by the two algorithms, in Figure 8, we only show the results coming from the approximation strategy (Algorithm 4). All above experiments are also carried out for $k = 10$, which gives similar results and thus is omitted here to save space. From Figure 8, we can see that for most utility measures, our backbone-based sampling approach can achieve good utility quality.

Note that our backbone-based sampling approach is a randomized strategy. One kind of aspects characterizing the efficiency of such strategies is its speed to converge to the steady state. To investigate the convergence of our approach, we need to explore the evolving trend of the aggregating statistics of utility measures with the increase of the number of graphs sampled. Here, for degree and shortest path length, we summarize the average of the value Kolmogorov-Smirnov statistic (which measures the maximum vertical distance between two cumulative distributions) as the difference between the original graph and the sampled graphs. The smaller this statistic is, the better the sampled graphs match the original graph on the compared distribution. We test the average value of this statistic on the two distributions considered, by increasing the number of sampling graphs from 1 to 100. Figure 9 gives the results.

As shown in Figure 9, in all the tested cases, the value of the utility measure used will fast converge to a steady value. And in many cases, only 5-10 sampled graphs are necessary to achieve relatively good utility quality. It's a strong proof of the efficiency and reliability of our sampling method. We thus could achieve a reasonably good approximation to the original graph's properties by sampling a very small set of subgraphs from the anonymized network.

5. IMPROVING K-SYMMETRY MODEL

In previous sections, we have shown that backbone-based sampling approach can ensure the utility of k-symmetry model, in spite of the fact that anonymization cost of the k-symmetry model is $O(|V(G)|^2)$. In some cases where minimization of modification on the original network is desired, reducing such costs will be a challenging problem. For this purpose, we will first propose a strategy based on accurate backbone computation to minimize the number of newly-introduced vertices in k-symmetry model. Then, we will improve the basic k-symmetry model by excluding the protection of hub vertices. We will elaborate the rationality and the benefits of our improvement.

5.1 Minimizing the Number of Newly-Added Vertices

A simple observation on Figure 3(a) inspires us that we can further reduce the number of newly-introduced vertices to satisfy the k-symmetry constraint. For example, in Figure 3(a), when $k = 3$, the orbit $V_1 = \{v_1, v_2\}$ needs to be copied once. However, this copy results in a new cell $V_1' = \{v_1, v_2, v_1', v_2'\}$ with four vertices, which is redundant for 3-symmetry (see Figure 5(b)). In fact, we just need to introduce only one new vertex v_1' to make the resulting cell $V_1'' = \{v_1, v_2, v_1'\}$ to be 3-symmetric.

Then an interesting question arises: *how to modify a network to be k-symmetric by introducing the* minimal *number of vertices?* Note that in Figure 3(a), v_1 and v_2 are automorphically equivalent to each other, which accounts for the redundancy of k-symmetry after orbit copying operations. Recall that a network's backbone is redundancy-free in the sense that the backbone cannot be obtained by applying any orbit copying operations on any of its subgraphs. Thus, if we can apply the anonymization procedure (Algorithm 1) on the network's backbone, i.e. $B_{G,Orb(G)}$ instead of the network itself, we can ensure the minimization of newly-introduced vertices.

5.2 Minimizing Anonymization Cost by Excluding Hub Protection

In this subsection, we will first discuss the motivation to exclude the protection of hubs in a social network and propose f-symmetry model as a generalization of k-symmetry model. Then we justify this strategy by experiments on real networks.

5.2.1 Anonymization Excluding Protection of Hubs

In this subsection, we will first show that *it is the hub vertices, i.e. vertices in the network with high degree, that dominate the anonymization cost of the k-symmetry model.* Heterogeneous degree distribution, the property shared by most of real world networks including social networks, states that most of vertices have small degrees and very few vertices have relatively larger degrees. Usually, we can easily identify those hub vertices from a social network. It has been shown that hub vertices tend to lie in a trivial orbit of a network [15], which implies that it's almost impossible to find an automorphically equivalent counterpart for a hub vertex. Such facts can be naturally interpreted since symmetry is sensitive to random perturbations of the network structure. Hence, to construct $k-1$ counterparts for each hub vertex, for example v, we need to introduce at least $k-1$ new vertices and $(k-1)deg(v)$ edges. Modification cost for constructing automorphic equivalence for a small number of hub vertices accounts for the majority of the overall modification cost to anonymize a network. Hence, if we improve our basic anonymity model by excluding the protection of some hub vertices, the modification on the original network

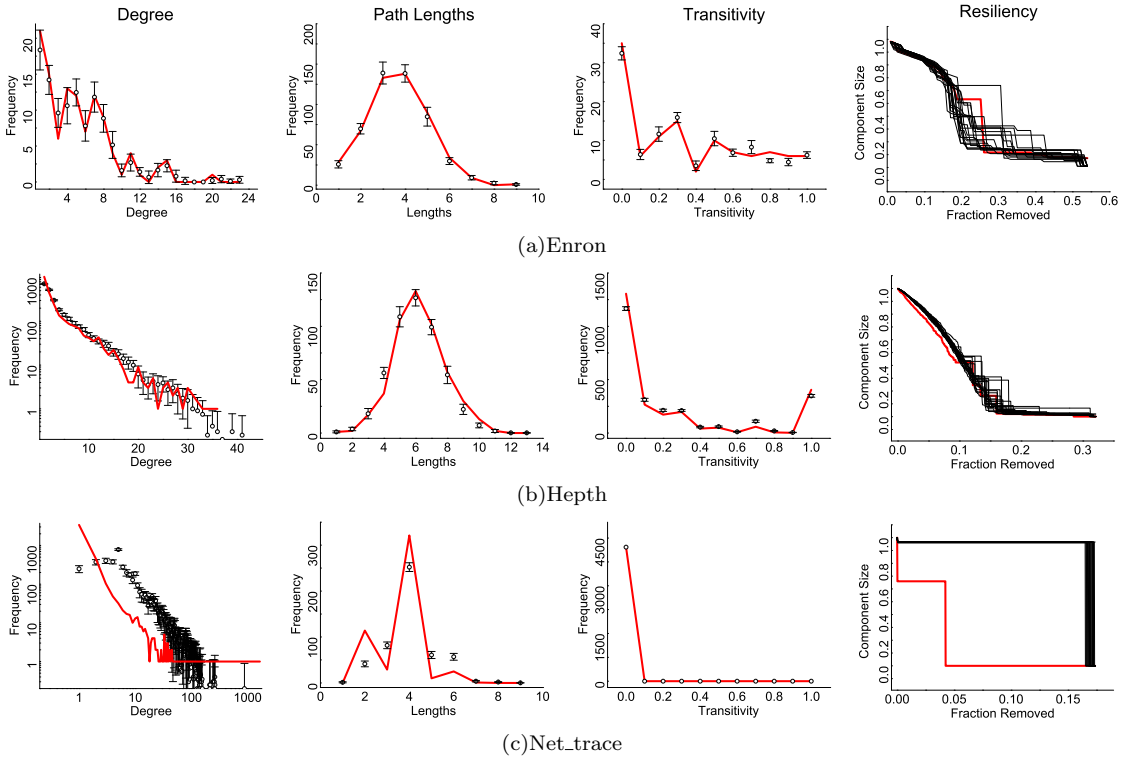
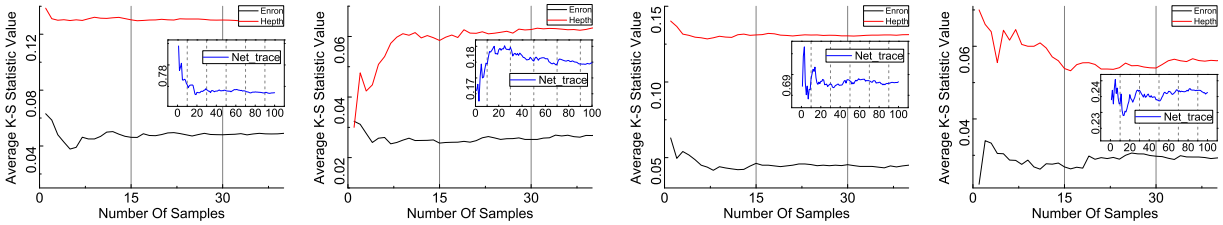


Figure 8: Experimental results of utility preservation. The figure compares sampled graphs computed by approximate backbone-based sampling algorithm (black) to the original graph (red).



(a) Degree Distribution ($k = 5$)
(b) Shortest Path Distribution ($k = 5$)
(c) Degree Distribution ($k = 10$)
(d) Shortest Path Distribution ($k = 10$)

Figure 9: Fast convergence of utility quality when the number of sampled graphs increases.

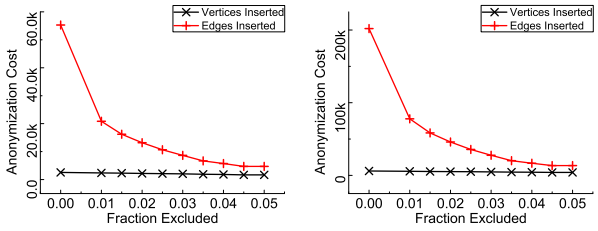
is expected to be significantly reduced.

When anonymizing social networks, we can find following obvious rationalities to exclude the protection of hub vertices. First, *in general, the hubs in a social network represent well-known individuals, for which identity protection is not necessary.* For example, in an email messaging network in a company, the most highly-connected vertex is quite likely to be the email address of the CEO. Hubs are often outliers in a network, making it difficult to protect their identity through anonymization [4]. Hence, we argue here that it is not necessary to protect hub vertices in identity anonymization for social networks. Second, *hub disclosure will not increase the risk of identity disclosure of other vertices and the link disclosure in the network.* Even the adversary knows the individuals represented by hubs, other vertices satisfying the k-symmetry constraint cannot be re-identified, and consequently any link in the network will be safe.

Above facts motivate us to improve the basic k-symmetry model to exclude the anonymization of hub vertices in a social network, which is defined in the following definition.

DEFINITION 5 (f-SYMMETRY). *Given a graph G and function $f : Orb(G) \rightarrow N$, if $\forall \Delta_i \in Orb(G) = \{\Delta_1, \Delta_2, ..., \Delta_m\}$, $|\Delta_i| \geq f(\Delta_i)$, then G is **f-symmetric**, or, G satisfies the requirement of **f-symmetry anonymity**.*

Clearly, k-symmetry model is just a special case of f-symmetry model, where f is defined to be a function mapping all the orbits in $Orb(G)$ to a constant integer k. Note that usually f is desired to be a non-increasing function with respect to d_i (degree of vertices in orbit Δ_i), that is, if $d_i \geq d_j$ then $f(\Delta_i) \leq f(\Delta_j)$. f-symmetry model provides further flexibility for the network publisher. The publisher could then test different functions and choose the one that can achieve the best utility results. As a real implementation of f-symmetry model, we may set a degree threshold δ and specify f to be a function that maps all the orbits containing



(a) $k = 5$ (b) $k = 10$

Figure 10: Anonymization cost when some hub vertices are excluded from protection.

vertices with degree above δ to 1 and maps the remaining orbits to k. Such a f-symmetry model can exclude the protection of hub vertices and consequently improve the utility quality.

5.2.2 *Experimental Results of Excluding Hubs*

In this section, we will show the benefits of excluding the anonymization of hub vertices by experimental results on the network **Net_trace**, whose degree distribution is extremely heterogeneous.

First, we investigate the relationship between the anonymization cost (quantified by the total number of new vertices and edges inserted) and the percentage of vertices not protected. As shown in Figure 10, when the fraction of vertices excluded (in the descending order of degree) increases slightly, the anonymization cost decreases dramatically. For instance, when $k = 10$, if 5% of vertices with largest degrees are excluded from protection, the number of inserted edges decreases from 201,913 to 13,444, saving nearly 94% overhead. What's more, even only 1% hub vertices are excluded from protection, we can save 61.5% overhead by decreasing the number of inserted edges from 201,913 to 77,749, which is an impressive achievement. From Figure 10, we also can see that usually the number of edges inserted dominates the overall cost.

We further explore the utility about the improved k-symmetry model by excluding some hub vertices. Intuitively, since less vertices and edges are introduced into the graph, the sampling approach can produce a graph approximating the original graph more closely. Figure 11 justifies this intuition. Here, we also use the average Kolmogorov-Smirnov statistic to measure the utility quality. We test this statistic on the degree and shortest path distribution for $k = 5, 10$. Since the fast convergence of this statistic has been verified, in this experiment, we simply summarize the statistic value for 100 sampled graphs. We highlight here that the anonymity power of the improved model will not be significantly degraded, since all vertices except some hub vertices still satisfy the k-symmetry anonymity requirement, and the number of hub vertices excluded is very small compared to the overall population.

6. RELATED WORK

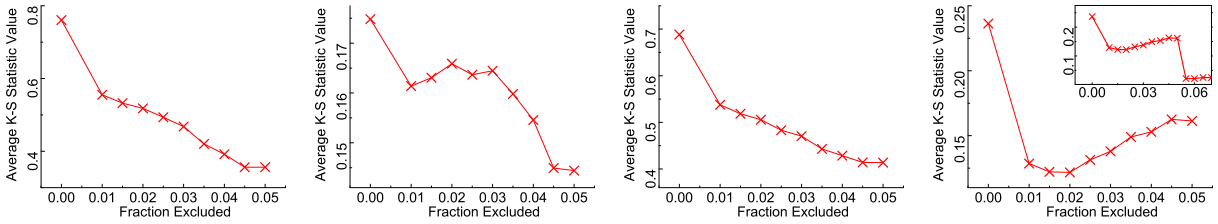
The problem of privacy protection in social networks was first proposed in [2], where the authors demonstrated that the naive anonymization strategy was not sufficient by studying both the active and passive model in depth. While active attacks are actually hard to carry out in many real social networks, however, passive attacks are much easier to do and thus have been more extensively studied. Some researchers focus on measures of anonymity (e.g., [12] and [14]), and others concern various anonymization techniques. In [5], a technique based on random edge deletions and insertions is proposed, which is effective to resist some kind of attacks but suffers a significant cost in utility. Edge randomization techniques are further explored in [18], whose goal is to preserve the spectral properties. While the network utility is much improved, the effect on anonymity is not quantified. Other anonymization techniques based on the classic framework of k-*anonymity*([11], [10] and [13]) which is widely adopted in the privacy preserving when releasing traditional tabular data, have also been proposed. Zhou et al. [19] introduce a method to insert edges into the network until any vertex has a local neighborhood which is isomorphic to at last $k-1$ other vertices. Liu et al. [7] present an efficient algorithm to make the network k-degree anonymous (i.e., for each vertex, there are at least $k-1$ other vertices sharing the same degree), also by inserting edges into the network. Most recently, Hay et al. [4] propose an anonymization technique which first partitions the vertex set into subsets with size at least k and then publishes a generalized network on the partition level.

Symmetry in real networks, which is fundamental to the k-symmetry model proposed in this paper, has only recently attracted research interests. It has been shown that various real networks have certain degree of symmetry [8, 17, 15]. Such symmetry can be produced by a network growth model following the principle called as "similar linkage patterns" [17]. If we collapse all structural redundancy characterized by network symmetry, we can obtain a structural skeleton of the parent network –network quotient, which preserves various key functional properties of the parent network [15]. Symmetry in real networks is further utilized to efficiently indexing shortest paths in a real network and answering queries on large graphs [16].

Quite recently, we notice that one of motivations in [20] is similar to us, which also *guarantees privacy under any structural attack*. However, the k-anonymity objective of [20] is k-*automorphism* and graph alignment is used to achieve this requirement. It will be an interesting future work to compare the efficiency and effectiveness of our approach achieving k-symmetry to that achieving k-automorphism. Whether k-automorphism is equivalent to k-symmetry still needs rigorous proof. However, some facts about k-symmetry can be given to help readers to differentiate it from other models: Given an integer $k > 0$, *if and only if for each vertex v in graph G, there exists $k-1$ nontrivial automorphisms (which means identity permutation is excluded) such that the images of any two of these automorphisms are distinct, then G is k-symmetric.*

7. CONCLUSION AND DISCUSSION

The major contributions of this paper can be summarized as follows. We study the extreme privacy protection problem in social networks: *protecting privacy against any possible SR*, and we propose k-symmetry model as an effective solution. In addition, we investigate the upper bound of the descriptive power of possible structural knowledge and quantify their power to re-identify a target. Furthermore, we efficiently implement the k-symmetry model, design two backbone-based sampling algorithms for utility preservation purpose, and conduct extensive experiments which demon-



(a) Degree Distribution ($k = 5$) (b) Shortest Path Distribution ($k = 5$) (c) Degree Distribution ($k = 10$) (d) Shortest Path Distribution ($k = 10$)

Figure 11: Utility improvements when excluding hub vertices.

strate both the efficiency and effectiveness of the proposed methods.

In our anonymization procedure, the automorphism partition of a graph is assumed to be given as the input. However, computing the automorphism partition of graph is not trivial, which is polynomially equivalent to the *graph isomorphism*(GI) problem [9]. In practice, program `nauty`[1] is usually used to compute the automorphism group of a given graph due to its computational efficiency. However, `nauty` may not scale well to large graphs with more than 20000 nodes. In such cases, a general approach called *graph stabilization* [6] may be used to produce a good approximation to the automorphism partition of the graph. One of such approximation is *total degree partition* $\mathcal{TDV}(G)$. We are surprised to find that for all the real networks that we've studied $\mathcal{TDV}(G) = Orb(G)$. Since approximation is acceptable for identity anonymization on a really large social network, $\mathcal{TDV}(G)$ will be a good substitute for $Orb(G)$ due to its computational efficiency.

8. ACKNOWLEDGMENTS

Authors would like to thank M. Hay for providing real network data. The work was supported by National Natural Science Foundation of China under Grants No. 60673133 and No. 60703093, National Grand Fundamental Research 973 Program of China under Grant No. 2005CB321905, Shanghai Leading Academic Discipline Project Under Project No. B114, Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant No. 200802461146.

9. REFERENCES

[1] R. Albert, H. Jeong, and A.-L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406:378, 2000.

[2] L. Backtrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In *WWW'07*, 2007.

[3] S. Fortin. The graph isomorphism problem. Technical Report TR 96-20, Dept. of Computing Science, University of Alberta, Canada, 1996.

[4] M. Hay, G. Miklau, D. Jensen, D.Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. In *VLDB'08*, 2008.

[5] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava. Anonymizing social networks. Technical Report 07-19, UMass Amherst, 2007.

[1] http://cs.anu.edu.au/~bdm/nauty/

[6] M. Klin and G.Tinhofer. Algebraic combinatorics in mathematical chemistry. methods and algorithms. iii. graph invariants and stabilization methods (preliminary version). Technical Report TUM-M9902, Technische Universitat Munchen, 1999.

[7] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD'08*, 2008.

[8] B. D. MacArthur, R. J. Sánchez-García, and J. W. Anderson. Symmetry in complex networks. *Discrete Applied Mathematics*.

[9] R. Mathon. A note on the graph isomorphism counting problem. *Information Processing Letters*, 8:131–132, 1979.

[10] P. Samarati. Protecting respondent's privacy in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 2001.

[11] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, SRI International, 1998.

[12] L. Singh and J. Zhan. Measuring topological anonymity in social networks. In *Intl. Conf. on Granular Computing*, 2007.

[13] L. Sweeney. k-anonymity: a model for protecting privacy. *Journ. of Uncertainty, Fuzziness, and KB Systems*, 2002.

[14] D. W. Wang, C. J. Liau, and T. S. Hsu. Privacy protection in social network data disclosure based on granular computing. In *International Conference on Fuzzy System*, 2006.

[15] Y. Xiao, B. D. MacArthur, H. Wang, M. Xiong, and W. Wang. Network quotients: Structural skeletons of complex systems. *Physical Review E*, 78:046102, 2008.

[16] Y. Xiao, W. Wu, J. Pei, and Z. H. W. Wang. Efficiently indexing shortest path by exploiting symmetry in graphs. In *EDBT'09*, 2009.

[17] Y. Xiao, M. Xiong, W. Wang, and H. Wang. Emergence of symmetry in complex networks. *Physical Review E*, 77:066108, 2008.

[18] X. Ying and X. Wu. Randomizing social networks: a spectrum preserving approach. In *SIAM Conf. on Data Mining*, 2007.

[19] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *ICDE'08*, 2008.

[20] L. Zou, L. Chen, and M. T. Özsu. K-automorphism: A general framework for privacy preserving network publication. *PVLDB*, 2(1):946–957, 2009.