Understanding the Mixing Patterns of Social Networks: The Impact of Cores, Link Directions, and Dynamics

[Last revised on May 22, 2011]

Abedelaziz Mohaisen Huy Tran Nicholas Hopper Yongdae Kim
University of Minnesota – Twin Cities
{mohaisen,huy,hopper,kyd}@cs.umn.edu

ABSTRACT

We extend upon our earlier results measuring the mixing time and advance our understanding of the problem in several directions. We relate the mixing time of social graphs to graph degeneracy, which captures cohesiveness of the graph. We experimentally show that fast-mixing graphs tend to have a larger single core whereas slow-mixing graphs tend to have smaller multiple cores. While this study provides quantitative and empirical evidence relating the mixing time to core structure of social graphs, it also agrees with our previous remarks on the potential relationship between the tight-knit community structure and slow mixing social graphs.

Equipped with with these findings, we advance this vein of research in several directions. First, we study several heuristics to improve the mixing time in slow-mixing graphs using their topological structures. Second, we study the impact of link direction and link dynamics on the quality of the mixing time. As anticipated, we find that dynamic social graphs have time-varying mixing characteristics. Counter-intuitively, we find directed graphs are faster-mixing than the same graphs when considered without edge direction.

1. INTRODUCTION

Leveraging social ties for building trustworthy computing services is becoming quite popular, and promises many primitives and applications for communication and security. Such applications and primitives benefit from both the trust exhibited in the underlying social networks—which rationalizes collaboration among nodes in the services built on top of it—and other algorithmic properties, which support the argument for the effectiveness of applications built on top of these networks. While most applications and primitives built on top of social networks share a commonality of purpose in using trust in the underlying graphs (e.g., for ensuring collaboration, rationalizing assumptions about the nature of the underlying social graph and attackers' capabilities, among many others), the algorithmic properties used in them differ greatly [30]. In the following, we elaborate on some of these exploited properties and their applications.

For example, the betweenness of nodes in social graphs—which captures how well-situated a node is on the path between other nodes—is used in [36] to build a Sybil defense in mobile networks. The betweenness and similarity are used in [6] for improving routing in delay tolerant networks. The closeness and connectivity are used in [13] for efficient content sharing and anonymity. Connectivity and transitive trust are used in [26] for improving Internet routing.

Another example of applications is social network based Sybil defenses, where several designs have been proposed to defend against the Sybil attack—an attack in which a misbehaving user claims multiple identities to influence the system's behavior [8, 23, 29, 36, 40, 46, 50, 51]. In most of these designs and defenses, trust is used to rationalize the difficulty of penetrating the social graph and establishing arbitrarily many links that thwart the utility of the defense mechanism, whereas the fast-mixing property is used to argue for the effectiveness of the detection of Sybil identities, and to produce a feasible solution. The same algorithmic property, of fast-mixing social graphs, is used in a closely related direction for demonstrating the utility of social graphs as good mixers for potential deployment of anonymous communication networks on top of them [34]. The same property is used in mobile networks for fast content dissemination [27].

Despite the importance of these properties to the usability of such applications, less effort is spent understanding the quality of such properties in real-world social graphs, and even less effort is spent relating these properties to other topological characteristics of social graphs. Motivated by the necessity of the assumption to operate these defenses, we have recently [33] measured the mixing time in several real-world social and information graphs and established that many social graphs are slower mixing than believed in the literature. We further demonstrated that the quality of the mixing time required for operating Sybil defenses for the majority of "honest" nodes in the social graph is less than assumed in the literature. At the down side, we have shown that some social graphs—despite the relaxation of the

required quality of the mixing time—have poor mixing characteristics which makes operating Sybil defenses on top of them infeasible.

1.1 Main Contributions

In this work, we extend upon our previous results and vastly advance our understanding of the mixing characteristics of social graphs in several directions, each of which is an interesting contribution in its own right. In the first direction, we re-examine the mixing characteristics of social graphs and relate the mixing pattern in such graphs to their topological structure. In particular, we relate the mixing time of social graphs to graph degeneracy, which captures the cohesiveness of the graph. We show that fast-mixing graphs are cohesive and consist of a single core even after the decomposition of the graph into its k-core [25] (a subgraph from the original graph obtained by iteratively pruning node with degree less than k). On the other hand, we show that slow-mixing social graphs tend to be less cohesive where they consist of multiple cores after the decomposition of their graph to the k-core—for relatively small k.

In the second direction, we investigate the use of graph structure to improve the mixing time of slow-mixing graphs. Arguing that many of the existing solutions in the literature for speeding up the mixing time are impractical to the context of many applications proposed on top of social networks, we consider several heuristics for improving the mixing time of slow-mixing graphs. Our heuristics are motivated by our findings on the structure of slow-mixing graphs and use auxiliary edges to "wire" separate cores. We investigate the potential of these techniques on the improvement of the mixing time. Counterintuitively, we show that the improvement of the mixing time does not depend on the number of edges used for connecting the separate cores but rather the way they are used: adding a small number of edges between cores improves the mixing time while adding more edges arbitrarily would reduce this quality.

In the third direction, we investigate the mixing time of *directed* graphs. Despite that many social graphs are directed by nature, they are converted to undirected graphs by completing their degrees [23, 50]. The main claim made in this context is that the majority of directed edges are in both directions between any pair of nodes, except in a small portion of nodes that break the symmetry of edges. However, the extent to which the modification of graphs in that way would affect the mixing time is not clear. Despite that the number of edges that break the symmetry of direction in edges could be as high as 50%, we unveil a surprising and a counterintuitive result by demonstrating that directed graphs are as good mixing as their undirected counterparts. In fact, we show two cases where undirected graphs are slower-mixing than the same graphs when considered with directions.

Finally, we formalize the problem of measuring the mixing time in dynamic graphs from an application point of view, and provide the sufficient mathematical tools to do that by extending upon the tools used for measuring static graphs. Using these tools, and two representative social graphs of dynamic structure, we measure the mixing time of dynamic graphs. As anticipated, we show that the mixing time of dynamic graphs is time-dependent. We further quantitatively measure how the variation in the mixing time of dynamic graphs would affect the operation of Sybil defenses.

1.2 Roadmap

The rest of this paper is structured as follows. In section 2, we review related work to motivate for our paper and multifold contributions. In section 3 we outline the preliminaries by reviewing theoretical tools required for measuring the mixing time in simple graphs, and show motivational examples on the mixing patterns of some real-world graphs. In section 4 we explore the relationship between the mixing time of social graphs and their core structures. In section 5 we explore methods for improving the mixing characteristics of slow-mixing social graphs motivated by our findings on the structure of these graphs. In section 6 we resolve the problem of direction in graphs by discussing tools for measuring the mixing time in a set of these graphs, and quantitively compare that to undirected graphs. In section 7 we explore the mixing time of dynamic graphs and show measurements of these graphs. Finally, in section 8 we outline future work directions and draw concluding remarks.

2. RELATED WORK

Using social networks to build primitives and services by exploiting social structures, social network properties, and trust in social graphs has been a growing area of research, where several directions are investigated to solve challenging problems. Social networks measurements is yet another active area of research, where most of the measured aspects in this paper have their share of interest in other contexts. Finally, improving the mixing characteristics of social networks, and formalizing the mixing characteristics of random walks in several graph settings have been actively investigated in the literature. Here, we elaborate on the relevant prior work in the literature and emphasize how it differs from our work. However, we limit ourselves to the domain of applications built on top of social networks for Sybil defenses.

2.1 Sybil Defenses Using Social Networks

While the idea of using social networks to improve security in distributed systems has been investigated

in [26] and [7], the first formal attempt to use social networks for defending attacks was in SybilGuard [51], which is uses fast-mixing graphs for Sybil detection. The same work has been extended and improved in SybilLimit [50]. Both SybilGuard and SybilLimit did not measure the mixing time, which is necessary for their operation. However, SybilLimit showed end-to-end results demonstrating that real-world social graphs mix well enough to support its requirements. Also using fast-mixing graphs, Danezis and Mittal introduced SybilInfer, an inference mechanism for Sybil nodes detection [8]. Tran et al. used the fast-mixing social graphs, and a ticket distribution mechanism that benefits from the mixing characteristics of social graphs, for building a Sybil resistant voting system called SumUp [46]. Lesniewski-Laas et al. [23] used the same property to build a Sybil-proof DHT system.

Based on the expansion which is related to the mixing time, Tran et al. introduced GateKeeper [45] which improves SybilLimit by reducing the number of sybil introduced per attack edge to $O(1)$. Using transitive trust and implicit assumptions on mixing characteristics of social graphs, Sirivianos et al. [40] and Mislove et al. [29] introduced two filtering algorithms of unwanted communications, both of which try to mitigate the impact of Sybil identities.

Though authors of most of these designs performed experiments to show insights on practicality of their designs, none of them measured the mixing time or any other property used for their operation directly from social graphs. Measuring these properties to understand the quality required for the operation of these designs is possible in most social graphs. This work is dedicated for understanding the main used property, the mixing time, under different scenarios.

2.2 Defense Analysis and Mixing Measurements

The use of assumptions in building Sybil defenses on top of social networks—like the "fast-mixing"—without verification motivated for investigating whether such systems work on various social graphs or not. Viswanath et al. [48] conducted an experimental analysis of Sybil defenses based on social networks by comparing different defenses (SybilGuard [51], SybilLimit [50], SybilInfer [8], and SumUp [46]). They demonstrated that different Sybil defenses work by ranking different nodes based on how they are well-connected to a trusted node. Also, they demonstrate that different Sybil defenses are sensitive to community structure in social networks and argue that community detection algorithms can be used to replace the random walk based Sybil defenses.

In [33], Mohaisen et al. measured the mixing time in several social graphs and demonstrated that social graphs are slower mixing than believed in literature. Furthermore, they noticed that mixing patterns in social graphs are associated with the underlying social model, where social networks with confined social models, and strict trust properties, are slow mixing whereas social networks with less strict trust properties are fast mixing. This observation is used in [30] to account for trust in social network-based Sybil defenses using modulated random walks.

Finally, Dellamico et al. [9] measured the mixing time in four datasets (Epinion, OpenPGP, DBLP, and Advogato) using the sampling method and the model in (3). Their work was motivated by evaluating the effectiveness of a wide range of designs based on both the mixing time and transitive trust. The authors *claimed* that real-world social networks meet theoretical assumptions of the Sybil defenses without showing that on any particular defense. The main conclusion made in [9] is that the mixing time is not associated with any of the known characteristics of the social graphs.

Our work in this paper is motivated by this part of related work, where our work advances existing findings vastly in many directions. We relate the mixing time to graph structure, propose heuristics to improve the mixing time based on this structure, measure the mixing time for naturally directed graphs, and formalize and measure the mixing time for dynamic graphs. In each of these directions we show several interesting findings based on our measurements.

2.3 Mixing and Graph Structure

To the best of our knowledge, there has been no prior work in the literature on understanding the mixing time of social graphs and relating it to graph structure, though there has been several works on understanding graph structural properties that could be indirectly related to the mixing time (such as diameter and clusters) [21, 22]. Thus our work is the first of its own type to consider this problem. On the other hand, controlling the mixing characteristics of social graphs, by either slowing them down or speeding them up, has been considered in several papers including [3, 11, 18, 30, 37]

In [37] it is shown that multiple random walks can be used to speed up the mixing characteristics of social graphs. In [11], a method is proposed for sampling multigraph, where it is claimed that these graphs are faster mixing than their simple counterparts. In [18] a heuristic is used to sample from graphs with respect to a certain biased distribution while maintaining a fast convergence rate to that distribution. In [3] a method for speeding up convergence to stationary distribution by providing nodes with random uniform jumps capabilities to the entire graph—which is in essence similar to PageRank [35] for directed graphs—is proposed and shown to speed up mixing characteristics of social graphs. Finally, in [30] several modulated random walks are proposed to control the mixing time (mostly by re-

ducing it) by limiting walks to trusted links, which in turn improves the operation of applications built on top of these graphs.

While these designs are of a great value, we argue that in many real-world scenarios they cannot be used directly to control the mixing time of social graphs due to their high cost and high risk to the utility of the intended applications by the improved mixing time. More details are in section 5.

2.4 Properties of Dynamic and Directed Graphs

Understanding dynamics in social and information networks has been extensively studied in the literature [4, 12, 16, 17, 41–44, 47, 49]. For example, in [12] Golbeck studied patterns of dynamics of social memberships and relationships. In [16], Kleinberg studied cascading behavior in networks in general, which applies to social networks under dynamics as well. In [4], Backstrom et al. studied group formation under dynamics in social networks. In [42], Tang et al. studied community evolution in dynamic social networks. In [43], Tantipathananandh et al proposed a framework for community detection in dynamic social graphs.

In [49], Wilson et al. studied interaction graphs by analyzing and comparing their structural properties, and measured the implication of their structure on applications, including Sybil defenses (SybilGuard in particular). While the results show significant trend of dynamics, no measurements for how these dynamics quantitatively affect the mixing characteristic of the social graph is provided. Similar to that, Viswanath et al., in [47], considered the (volume-wise) evolution of interactions among users in Facebook, but did not measure the mixing time of the time-varying social graph. In [30] we measured the mixing time of an interaction graph borrowed from [49] and compared it to the relationship-based graph. We demonstrated that the former graph is slower mixing than the latter one, but did not consider the problem of measuring the mixing time of the evolving graph at different times due to the lack of data and tools.

Findings in most of these works are different in essence. However, they convey similar conclusions by pointing out the impact of dynamics on measured characteristics in social graphs. Yet, none of these works considered how these dynamics affect the mixing time. In this work, we go one step farther by formalizing time-evolving graphs, define their mixing time, and measure it in two real-world traces.

Measuring the mixing time in directed graph is not explored yet too, though there has been some recent theoretical efforts on formalizing the problem of fast-mixing directed graphs, as in [15]. In this paper, as the first result in that direction, we elaborate on formalizing the mixing time of directed graphs, show how to measure it, and measure it for several directed graphs. We compare the measurements to undirected graphs derived by omitting graph directions.

3. THE MIXING TIME IN SIMPLE GRAPHS

Here we elaborate on the method used in measuring the mixing time in undirected unweighted (simple) graphs, which is simplest to consider. We begin by formalizing the model of these graphs, the mixing time, and some motivational measurements borrowed from our prior work in [33].

3.1 Graph Model and Uniform Walks

Let $G = (V, E)$ be a simple undirected and unweighted graph, where V is the set of vertices of G such that $|V| = n$ and E is the set of edges in G such that $|E| = m$. For G, we define $\mathbf{A} = [a_{ij}]^{n\times n}$ as the adjacency matrix, where

$$a_{ij} = \begin{cases} 1 & v_i \sim v_j \\ 0 & \text{otherwise} \end{cases}. \quad (1)$$

($v_i \sim v_j$ means that v_i is adjacent to v_j). Let the number of nodes in V adjacent to v_i be $\deg(v_i)$ (also computed from $\mathbf{A}$ as $\deg(v_i) = \sum_j a_{ij}$). For G, we define the stochastic transition probability matrix $\mathbf{P} = [p_{ij}]$ of size $n \times n$ where the $(i, j)^{\text{th}}$ entry in $\mathbf{P}$ is the probability of transitioning from node v_i to node v_j in one step, which is defined as follows

$$p_{ij} = \begin{cases} \frac{1}{\deg(v_i)} & v_i \sim v_j \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Let $\mathbf{D} = [d_{ij}]^{n\times n}$ be a diagonal matrix, where $d_{ii} = \deg(v_i)$, then the transition probability is computed as $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$.

3.2 The Mixing Time

Moving from a node to another on G is captured by the Markov chain which represents a random walk over G. A random walk of length t over G is a sequence of vertices in G such that each node is selected at its predecessor in the random walk following the transition probability defined in (2). The Markov chain is said to be *ergodic* if it is irreducible and aperiodic, where in such case it has a unique stationary distribution π and the distribution after t steps converges to π. The stationary distribution of the Markov chain is a probability distribution that is invariant to the transition matrix $\mathbf{P}$ (i.e., $\pi\mathbf{P} = \pi$). The mixing time, T, is defined as the minimal length of the random walk to reach the stationary distribution. More precisely, the mixing time of a Markov chain on G parameterized by a variation distance ϵ is defined as

$$T(\epsilon) = \max_i \min\{t : |\pi - \pi^{(i)}P^t|_1 < \epsilon\}, \quad (3)$$

where $\pi = [\deg(v_i)/2m]^{1\times n}$ (for simple graph), $\pi^{(i)}$ is the initial distribution concentrated at vertex v_i, P^t is the transition matrix after t steps, and $|\cdot|_1$ is the total variation distance, defined as $\frac{1}{2}\sum_{j=1}^{n}|\pi_j^{(i)} - \pi_j|$. We say that a Markov chain is "fast-mixing" [8, 23, 50, 51] when $\epsilon = \Theta(\frac{1}{n})$, and $T(\epsilon) = O(\log n)$. An interesting result on bounding the mixing time for simple graphs is provided in [39], and used in [33] for measuring the mixing time of several real world social graphs. In short, the method uses the second largest eigenvalue, μ for bounding the mixing time. The closer μ is to 1, the slower the mixing time, and faster otherwise.

Measuring the mixing time can be done using both definitions [33]. However, since using the second largest eigenvalue would account only for the poorest mixing source in the social graph, measuring the mixing time using the definition in (3) is highly desirable to express the richer patterns of mixing, which represent various initial sources in the social graph. In [33], we used this observation to capture the variety of mixing patterns in the same social graph. By varying sources of walks, and sampling such sources from G, one can get a good estimate about the distribution of the mixing across different sources, and thus the entire graph. By definition, however, the mixing time of the graph is defined as the mixing observed at the slowest source, thus every other source in the graph has at least that mixing bound.

Table 1: Datasets of (undirected) social graphs for which we measure the mixing time.

Dataset	# nodes	# edges
Youtube [28]	1,134,890	2,987,624
DBLP [24]	769,641	3,051,127
Slashdot [22]	70,355	459,620
Epinion [38]	32,223	342,012
Physics 2 [20]	11,204	117,619
Gnutella [20]	4,317	18,742
Physics 1 [20]	4,158	13,422
Wiki-vote [19]	1,300	36,529

3.3 Motivational Initial Results

We borrow the same tools in [33], which are explained section 3.2, to measure the mixing time of social graphs and highlight the variability in mixing patterns across different social graphs. Datasets used for this measurement are shown in Table 1. For directed graphs (4 graphs in Table 3), we use the graph conversion method in section 6.2.1. More details on these graphs are in [33] and the corresponding citation along with each graph. From each of these graphs, we select 1,000 initial nodes uniformly at random and compute the statistical distance ϵ as we increase the length of the random walk t. At each time step we compute the basic statistics of ϵ—the max, mean, and median. Results of this measurements are shown in Figure 1, where the main conclusion [33] is that social graphs differ in their mixing characteristics, independent of their size and density. This conclusion can be made clear on Figure 1(b) (and on 1(a) and 1(c) for max and median with slight differences) by observing that while Gnutella's number of nodes is 3 times larger larger than that of Wiki-vote, and while Wiki-vote is 3 times denser than Gnutella, Gnutella still has better mixing characteristics than any other graph we tested. Another example is by observing Youtube and Physics 2. Physics 2 dataset is three times denser and 100 times smaller, yet it has same mixing characteristics as Youtube. Other examples can be easily driven from these results to demonstrate this tendency.

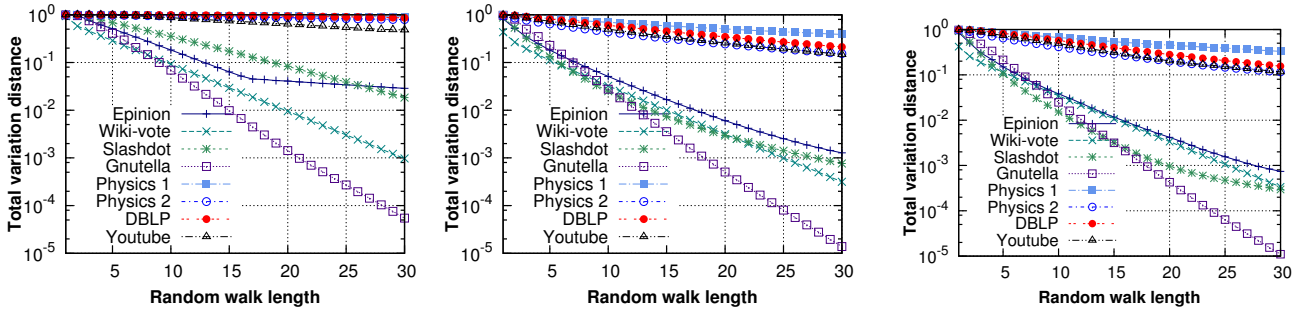
An interesting question that rises now is: *can we relate the mixing time in these graphs to their structure?* What is obvious is that simple characteristics, like density and size, are not be sufficient to answer this question. Moreover, degree distribution, clustering coefficient, and others are not sufficient to understand the mixing pattern as shown in [9]. It is intuitive to think that fast-mixing graphs should have inherit property in their structure, which captures their "connectivity". The connectivity itself is related to the mixing time.

For example, the conductance which is an indicator of how well-connected is a graph gives a lower bound of the connectivity and mixing time. However, computing the optimal conductance is NP-hard [2] and known approximation is a $O(\log n)$-approximation which runs in $O(n^2)$ [1], making it expensive to compute for large graphs. Also, even if the exact conductance is computed, it would be used for reasoning about the lower bound of the mixing time, but not the general pattern in the graph. Combining that with our findings in [33]—where we show that some slow-mixing graphs according to the definition are slow due to a small portion of nodes—would make the use of conductance to understand the mixing pattern in social graphs of less interest.

We explore another graph structure property that could sufficiently distinguish slow-mixing from fast-mixing graphs and show the pattern in their structure. More importantly, this property can be computed in $O(n)$ for an exact solution, and can be done for large (million nodes) graphs feasibly.

4. THE MIXING AND GRAPH STRUCTURE

It is unclear what properties slower mixing graphs possess. Furthermore, it has been claimed in the literature that the mixing time do not relate to any of the graph structure properties, making the mixing time interesting in its own right [9]. Here, we re-examine this statement by trying to understand the mixing characteristics of social graphs and relating them to graph structure. Indeed, we find that mixing characteristics



(a) Max (b) Mean (c) Median

Figure 1: A measurement of the mixing time of undirected graphs shown in Table 1 [Best viewed in colors].

are very related to core structure of the social graph, which captures graph cohesiveness. We show that fast mixing graphs tend to have large *single* core, whereas slower mixing graphs tend to dissolve into multiple cores. Before getting into the details of our measurements, we lay down the preliminaries by defining graphs cores.

4.1 Graph Cores and Node Coreness

Let $G = (V, E)$ be a graph defined according to the same model in section 3. For any k where $1 \leq k \leq k_{\max}$, let $G_k = (V_k, E_k)$ be a subgraph in G such that $V_k = n_k$ and $|E_k| = m_k$, and with the constraint that for all $v_i \in V$, the minimum degree of any node $v_j \in V_k$ is k. G_k is said to be a $k-$core of G if, in addition to the above condition, it is a maximal and connected graph. If we relax the connectivity condition of the $k-$core, we get a set of cores (potentially more than one) where each of such cores satisfy the degree condition. Such set of cores is represented as a (disconnected) graph $G'_k = (V'_k, E'_k)$ where $|V'_k| = n'_k$ and $|E'_k| = m'_k$. An efficient algorithm for decomposing a simple graph to its $k-$cores by iteratively pruning nodes with degree less than k has the complexity of $O(m)$ where m is the number of edges in the graph [5]. We define the node-relative (or normalized) size of G'_k as n'_k/n and the edge-relative size of G'_k as m'_k/m. Similarly, for G_k (the largest core) we define these normalized quantities as n'_k/n and m'_k/m. Finally, for every node $v_i \in V$, we define the coreness c as the largest c s.t. $v_i \in G_c$ where G_c is a $c-$core.

The definition of k-core and graph coreness [25] is equivalent to many other interesting graph properties, such as k-coloring [10]. In particular, a k-core graph is $(k+1)$-colorable, which does not only depend on the density of the graph but the way edges are established between nodes. Indeed, we show in section 5 that increasing the density of the graph arbitrarily does not necessarily improve the mixing time.

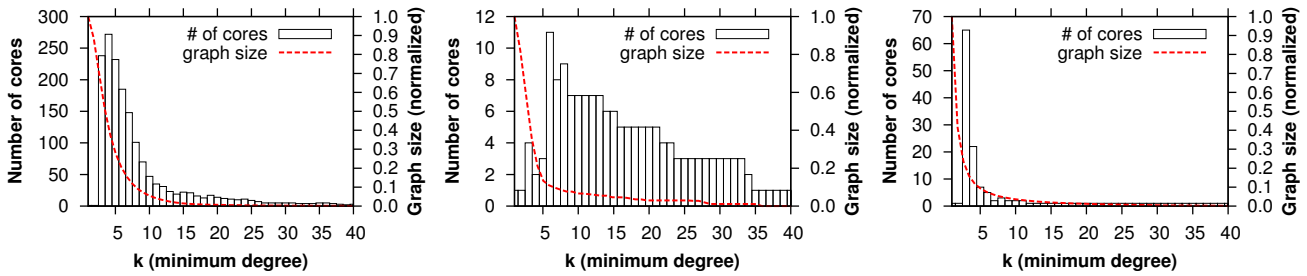
4.2 Measurements and Results

We use the same datasets shown in Table 1 to explore the pattern of mixing time in social graphs by decomposing them to their core structures defined above.

For each of these graphs we use an off-the-shelf implementation of the linear-time algorithm in [5] to compute the $k-$core by relaxing the connectivity assumption, which could lead to a disconnected graph. As k increases to its ultimate value at which the graph disappears (no node in it would have a c-coreness equal to that ultimate k) we compute the following: (1) the number of cores in the k-core as we decompose the graph for increasing k, (2) the size of each k-core as we increase k normalized by the original graph size, and (3) the size of each core in the k-core as we increase k. Among these we are in particular interested in the first and second metrics—the number and normalized total size of the cores in the k-core. The results of these measurements shown in Figure 2 and Figure 3. Notice that graphs in Figure 2 are slow mixing and graphs in Figure 3 are fast mixing, as shown in Figure 1. We make two observations on these results that capture their fundamental differences.

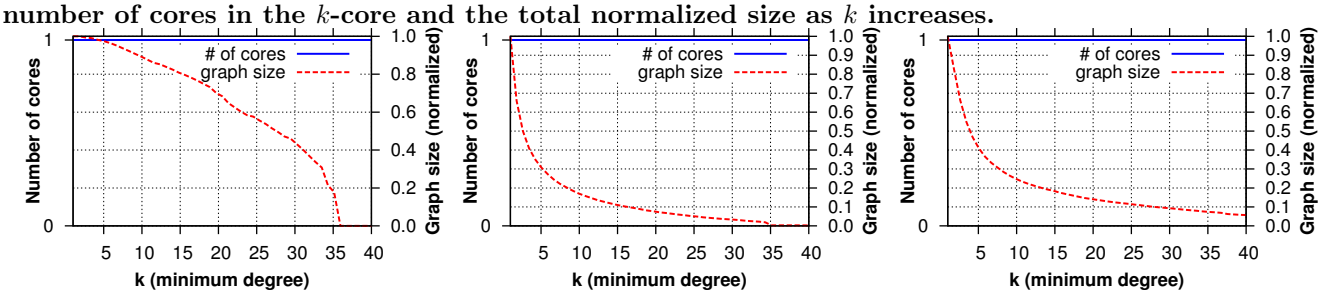
First, the most obvious result by comparing Figure 2 to Figure 3 is that slow mixing graphs are less cohesive whereas fast mixing graphs are more cohesive. This cohesiveness is reflected on the number of cores in the k-core of each graph as we increase k. Whereas slow mixing graphs shown in Figure 2 are decomposed into multiple cores as we increase k, fast mixing graphs resist this decomposition and remain cohesive as we increase k (shown in Figure 3).

Second, despite that slow mixing graphs are decomposed into multiple cores, these cores are relatively small in size and the disappearance of the graph is very quick in many cases as we increase k; see for example Figure 2(a)) and Figure 2(c). On the other hand, though they have single core for an increasing k, they remain mostly large in size and resist dissolution; see for example Figure 3(a) and Figure 3(c). While the general tendency in the decrease of the normalized graph size as we increase k is reciprocal in the order of k, with the difference being a constant among graphs, we observe some of the fact mixing have a negative exponential order of k (e.g., $-c_1 e^k + c_2$ for two positive constants c_1



(a) DBLP (b) Physics 1 (c) Youtube

Figure 2: A measurement of the core structure in a select of social networks, representing both the number of cores in the k-core and the total normalized size as k increases.



(a) Wiki-vote (b) Slashdot (c) Epinion

Figure 3: A measurement of the core structure in a select of social networks, representing both the number of cores in the k-core and the total normalized size as k increases.

and c_2). This latter case is illustrated in Figure 3(a) where the decreases in the graph size is very small.

Limitations of the measurements: it is obvious that our measurements only consider two extremes: fast mixing graphs and slow mixing graphs but nothing in between. In reality, graphs tend to have mixing patterns that can be differentiated at average (or at maximum as per the definition of the mixing time). These differences are not captured here by the method we considered, though we anticipate that this can be related to the two metrics we have considered here. Further theoretical and empirical investigation on this issue would be an open direction to consider in the future.

Now that we established that there is a fundamental difference between fast and slow mixing graphs in terms of their structure, can we use these differences for slow mixing graphs to improve their mixing characteristics? We try to answer this question in the following section.

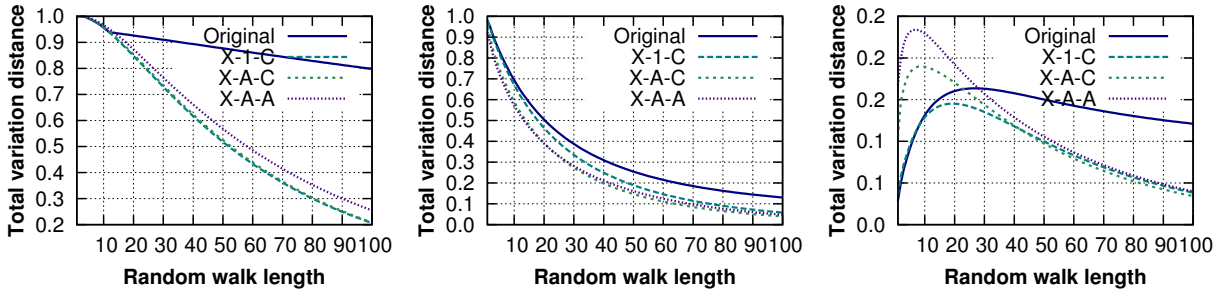
5. IMPROVING THE MIXING TIME

Now that we learned the fundamental difference in structure between fast and slower mixing social graphs, a natural question that rises is how to improve the mixing time of slower mixing social graphs. Indeed, there has been several successful attempts in the literature to improve the mixing characteristics of Markov chains on graphs [3, 11, 14, 15, 35, 39]. A well-known example of such attempts is using teleportation probability [3] (which borrows ideas from [35]). Each node in the graph decides to follow the uniform random walk protocol by selecting the next hop of the walk from its direct neighbors uniformly at random, with a total probability $1-\alpha$, or direct the random walk towards a uniformly chosen node at random from the rest of the graph with a total probability α. α is chosen to be small enough so as not to deviate or change the overall random walk structure and the final bounding or stationary distribution.

5.1 Limitations of Existing Solutions

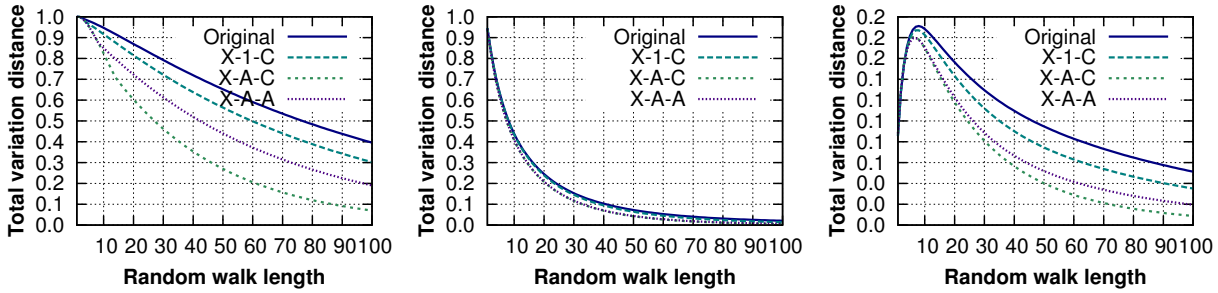
While such scheme in theory would work and provide the intended guarantees in improving the mixing time, it comes at high cost in practice [3]. Imagine an application like a Sybil defense that utilizes social graphs for its bootstrapping, or an anonymous communication system that uses social links, or information dissemination algorithm that forwards information over social links, among many others. In each of these applications, users are limited by the number of nodes with whom they share relationships, and if such "teleportation" process existed it would be with cost [3], which would be high in many cases. Furthermore, incentives may not exist for using such scheme at scale.

Another fundamental reason why such an algorithm for improving the mixing time would not be practical is that it improves the mixing time of nodes regardless to their labels: honest or dishonest. This in fact is due to that a node would not be able to verify whether a far away node that forwards the random walk towards it through those jumps is honest or not, since most likely both nodes would be separated by multiple hops. In a Sybil defense that uses social networks, while one would be interested in improving the mixing time of the honest region of the social graph, Sybil (dishonest) nodes in the graph also would benefit from the teleportation probability used for performing the uniform jumps in



(a) Max (b) Mean (c) Standard Deviation

Figure 4: Mixing time measurement of Physics 1 before/after using the heuristics to improve its mixing characteristics.



(a) Max (b) Mean (c) Standard Deviation

Figure 5: Mixing time measurement of Physics 2 before/after using the heuristics to improve its mixing characteristics.

the random walk, thus violating the basic assumption of "sparse-cut" used in the defense for identifying dishonest nodes.

To this end, we outline two requirements for an ideal algorithm that improves the mixing time in order to be sufficient for this type of applications. First, such an algorithm should improve the mixing time of the good nodes in the social graph. Second, the same algorithm should maintain the property of the social graph which differentiates honest from dishonest nodes. As in above, the algorithm should not improve the extent to which random walks originated from the dishonest region mix with the honest part of the graph.

5.2 Heuristics to Improve the Mixing Time

To achieve both goals, using the structural property of slow-mixing social graphs, we propose several heuristics. In the $k-$core graph decomposition process, a graph G_i is decomposed into its G_{i+1} core by recursively trimming all of its nodes with a degree less than or equal to i. By doing so, we have observed that the graph G_{i+1} is decomposed into multiple cores; one of these cores is the major core (largest in size), while other cores are relatively smaller in size (minor cores). While it might be impractical, especially for smaller i, to postpone or prevent the decomposition of the graph, one could try several techniques, which can be turned into algorithms to postpone or prevent the decomposition of the graph into multiple disconnected components. One direct approach is to connect (or *wire*) disconnected cores in the graph G_i to each other using *auxiliary edges*. This indeed leads to the following three scenarios.

Scenario X-1-C: wire a single node in each minor core X with a node in the major core C using *one* edge. The end vertices of added edges can be arbitrarily chosen. By doing so, we can easily see that the graph will always have a single core at any time when increasing k. The number of added edges is the sum of the number of cores in each k-core, for all k, minus k itself (to exclude major and single cores).

Scenario X-A-C: wire each node in each of the minor cores with a node in the major component, as we increase k. Same as above, this would prevent producing multiple cores at time and the number of auxiliary edges is bounded by the *number of nodes* in the minor component.

Scenario X-A-A: wire all nodes in a minor core to other cores in the graph, including both minor and major cores. The number of auxiliary edges is bounded by the *order* of the number of nodes in each k-core.

Notice that in each of these scenarios, we need to improve the mixing time of the honest region of the graph, so wiring a fixed number of *honest nodes* is ideally performed using this method. In our study, and for the limitations of our datasets, we consider that the entire graph is of honest nodes to derive insight on the potential of each method in improving the mixing time. Extending this to the general case when labels of nodes are given is straightforward. Furthermore, when considering wiring unlabeled nodes even if an edge is cre-

ated between an attacker and an honest node, it would have a limited impact on the number of Sybil identities introduced per that attack edge, as in SybilLimit for example.

The rationale of using auxiliary links is very natural and simple. Such links can be either virtually created between nodes if the application that uses the social graph is centralized [46], or through link recommendations if this process is to be performed in a decentralized application [50].

5.3 Results and Discussion

We select two of the slow-mixing and relatively small social graphs to explore the potential of our heuristics. These graphs are shown in Table 1 with their initial statistics. The results of measuring the mixing time after applying the heuristics in section 5.2 and for the same settings of measurements as in section 3 are shown in Figure 4 and Figure 5, respectively. The total number of edges before and after wiring graphs according to the methods in section 5.2 is shown in Table 2. On these graphs we make following observations.

First of all, our simplest heuristic (X-1-C), which produces minimal effect on the graph density—122 edges are added to Physics 1—significantly improves the mixing time according to its definition as the maximal t for a given ϵ and provides a correction on the slowest mixing sources. This source, by definition, would have the poorest mixing characteristics resulting in a poor mixing in the overall graph as shown in Figure 4(a). Similar improvement is obtained on average for mixing time, as shown in Figure 4(b). The standard deviation among all source for which the mixing characteristics are computed is also being improved as the addition of these few auxiliary edges regulate the structure of the graph, as shown in Figure 4(c).

Second, the extent to which additional edges improve the mixing time differs and depends on the initial mixing characteristics of the graph. Graphs that mix better than others (yet overall slow-mixing) tend to have less number of cores which means less number of additional useful edges are added to these graphs. For example, X-1-C adds about 68 edges to the original Physics 2 graph, which translates into less effect on the mixing time, as shown in Figure 5. The original graph already mixes better than Physics 1 dataset (on average) and the addition of these edges, though improves the slowest mixing sources (and the mixing of the graph as per the definition), it does not improve a lot on average. For the same reason, better improvement on average is realized in Physics 1 as shown in Figure 4.

Third, by considering the number of added edges in X-A-A in both datasets and the measured mixing time, we observe that the addition of a lot of edges—despite improving the density of the graph—does not improve the mixing time. Indeed, we find that using this heuristic would slow the mixing of these graphs down. One potential reason for this behavior is that, as we add more edges, random walks are diverted from traversing other nodes in the graph by the additional edges connecting minor cores. This diversion is translated into slower convergence to the stationary distribution.

Table 2: The number of edges in the different social graphs before and after wiring them according to our heuristic for improving their mixing characteristics. The number of nodes is shown in Table 1.

Dataset	Number of edges (total)			
	Orig.	X-1-C	X-A-C	X-A-A
Physics 1	13422	13544	16482	25064
Physics 2	117619	117687	119082	121169

6. THE IMPACT OF LINK DIRECTION

Many graphs in general, and social graphs in particular, are directed by nature. On the other hand, many applications, including Sybil defenses, information routing, dissemination, and anonymous communication—when built by exploiting social structures—require mutual relationships which produce undirected graphs. However, when undirected graphs are used as testing tools for these applications to bring insight on their usability and potential, directed graphs are transformed into undirected graphs by omitting directions. One claim made in that context is the percent of added edges to complete edges in both directions and make them undirected is small (e.g., 10% of the total number of edges among all nodes), and thus converting the entire directed graph to undirected would not incur a major impact on the graph structure. Whether this is the case or not when it comes to the mixing time of the social graph is not tested before, though the intuition is that directed social graphs would have different, and likely slower, mixing patterns than undirected graphs. Here, we investigate this issue and qualitatively measure the difference in the mixing time in both cases. Before introducing the measurements, we elaborate on the way to measure the mixing time in directed graphs and how this differs from the undirected case.

6.1 Mixing Time of Directed Social Graphs

Conditions required for defining the mixing time on undirected and directed social graphs, and the way used for computing the mixing time, are slightly different. While connectivity is required in both cases to define the mixing time, the directed graph requires this connectivity in the form of a *strongly connected component* (SCC). In SCC there exists a path between every pair of nodes in the graph. Computing the SCC of a graph in

linear time can be done using Tarjan's algorithm, which we use in this work for different graphs.

Notice that obtaining the SCC from the directed graph is necessary for defining the mixing time. From an application point of view, if a design like a Sybil defense is to be used on top of a directed graph then either that graph needs to be SCC at the first place or to have a fixation for the isolated components in that graph so the entire graph is an SCC. One potential way for fixing this problem to graphs that are not SCC in their entirety is to use our mixing time improvement method by wiring honest nodes in two disconnected components so as to allow flows in the entire graph. We leave this direction as a future work, and limit ourself to the SCC in these graphs which could potentially not include all nodes.

While the definition of the mixing time in both directed and undirected graphs is still the same, the bounding (stationary) distribution of Markov chains defined on both graphs is different. In particular, unlike the undirected graph where we can easily express the stationary distribution in a clean form in terms of node degree as shown in section 3, the stationary distribution in case of the directed graph does not relate to node degree (not the in-degree nor the out-degree).

6.1.1 Defining the Mixing Time for Directed Graphs

Similar to the model in section 3, let $\mathbf{A} = [a_{ij}]^{n\times n}$ be the adjacency matrix of a directed graph $\vec{G}$ where $a_{ij} = 1$ if an (directed) edge exists from node v_i to node v_j (denoted by $v_i \rightarrow v_j$), and 0 otherwise. Notice that $\mathbf{A}$ is most likely asymmetric, unlike in undirected graphs. Let $\deg(v_i)^-$ be the out-degree of node v_i. We define the transition probability matrix $\mathbf{P} = [p_{ij}]$ where $p_{ij} = 1/\deg(v_i)^-$ iff $v_i \rightarrow v_j$ and 0 otherwise. In a clean matrix form, we get $\mathbf{P} = (\mathbf{D}^-)^{-1}\mathbf{A}$, where $\mathbf{D}^-$ is a diagonal matrix with the ii-th element in it defined as $\sum_j a_{ij}$. We define the stationary distribution as $\pi = J\mathbf{P}^t$ (which could be used for any connected graph), where $t \rightarrow \infty$ and J is a $(1 \times n)$-vector in which all entries are ones. As t grows, one would lose the sparsity of $\mathbf{P}$. Thus, computing the stationary distribution using the method above is time and space inefficient.

6.1.2 Estimating the Stationary Distribution

Here we devise a method for computing a good estimate of the stationary distribution without having to modify the structure of $\mathbf{P}$ which leads to inefficiency. We observe that, regardless of the initial distribution, every walk on a strongly connected graph would ultimately converge to the stationary distribution π. We could further make the total variation distance between the ideal stationary distribution and the accumulated distribution of a random walk beginning from an arbitrary node in the graph arbitrarily small. In other words, given an arbitrary initial distribution $\pi^{(*)}$ and the matrix $\mathbf{P}$, we can compute $\pi_\ell^{(*)} = \pi^{(*)}\mathbf{P}^\ell$ so as $|\pi_\ell^{(*)} - \pi|_1 < \delta$ where δ is very close to 0, for some large walk length ℓ.

The convergence of $\pi_\ell^{(*)}$ is guaranteed by the definition of the stationary distribution. Without knowing the (ideal) stationary distribution π, one could use $\pi_\ell^{(*)}$ as an estimate for π, for *large* ℓ. Such distribution is sufficient to measure a very close estimate of the mixing time of the directed graph in an efficient way, and to serve the purpose of our measurement in understanding the difference between the mixing patterns in directed and undirected graphs under similar circumstances. In [32] we bound the error between these measurements and ideal measurements to a small negligible constant. Furthermore, to speed up the convergence to π, we begin from a uniform distribution $\pi^{(*)} = [1/n]^{1\times n}$.

Notice that computing $\pi_\ell^{(*)} = \pi^{(*)}\mathbf{P}^\ell$ does not require computing $\mathbf{P}^\ell$. We can iteratively compute $\pi_\ell^{(*)}$ using a vector-matrix multiplications of $\pi^{(*)}$ and $\mathbf{P}$, where we can benefit from the sparsity of $\mathbf{P}$. In our measurements, and to make $\pi_\ell^{(*)}$ a good estimate of π we set $\ell = 10,000$, for which we observe a steady distribution with no significant difference between $\pi_{\ell-1}^{(*)}$ and $\pi_\ell^{(*)}$ as ℓ increases.

6.1.3 Computing the Mixing Time

The remaining part of computing the mixing time of directed graphs is straightforward. Given $\pi_\ell^{(*)}$ and $\mathbf{P}$, we define the mixing time as follows (similar to Eq. 3, see Appendix A in [32] for bounding the error in this definition).

$$T(\epsilon) = \max_i \min\{t : |\pi_\ell^{(*)} - \pi^{(i)}\mathbf{P}^t|_1 < \epsilon\},$$

Using the same method as when computing the mixing time of the undirected graph, we begin by a sample of initial distributions (of nodes) uniformly selected at random and compute ϵ as above for an increasing t. The mixing time as by definition is the maximum t for a given ϵ, while other statistics such as the mean and standard deviation are good indicators of the general tendency of the mixing pattern.

6.2 Results and Discussion

Here we outline the main results and findings of measuring the mixing time of directed graphs. We compare our findings to the original method in which directed graphs are converted into undirected graphs. This method is previously used in [33] for measuring the mixing time, and follows the same method of graph converging as in [23] and [50].

6.2.1 Datasets and Data Preprocessing

Four different datasets are used for this measurement and are shown in Table 3. Three of these datasets are

Table 3: Datasets used for measuring the mixing time in directed graphs with their statistics. The undirected graph is obtained from the SCC and the difference is computed between $2m$ of undirected graph and m in directed graph.

Dataset	Original graph			Largest SCC		Largest SCC %		Undirected (SCC)		Difference	
	n	m	# SCC	n	m	n (%)	m (%)	n	m	# edges	percent
Slashdot	77,360	905,468	6,724	70,355	818,310	90.94	90.37	70,355	459,620	100,930	10.98
Epinion	75,879	508,837	42,176	32,223	443,506	42.47	87.16	32,223	342,013	240,520	35.16
Wiki-vote	7,115	103,689	5,816	1,300	39,456	18.27	38.05	1,300	36,529	33,602	45.99
Gnutella	10,876	39,994	6,560	4,317	18,742	39.69	46.86	4,317	18,742	18,742	50.00

previously used for measuring the mixing time in [33], and represent social networks. The last dataset (Gnutella) [21] is of a peer-to-peer system for file sharing, where nodes are hosts and edges indicate that a host is connected to another host.

The first dataset is of the Slashdot Zoo [22], in which a directed edge between two nodes indicates that the first node tags the second node as a friend. The second dataset is of Epinion [38], a who-trust-whom online social network. An edge between two nodes indicates that the first node has tagged the second node as a trusted node. The third dataset is of wiki-vote [19], which contains voting for wikipedia administrators promotion. A link between two nodes indicates that the first node has voted for the second node.

For each of these graphs, in order to satisfy the connectivity condition required for measuring the mixing time, we compute the largest strongly connected component (SCC). The SCC of each graph and its relative size compared to the original graph is shown in Table 3. The largest SCC varies in size, and ranges from as low as 18% of the number of nodes in the original graph (as low as 38% of edges, as in Wiki-vote) to as high as 90% of nodes (and edges, as in Slashdot).

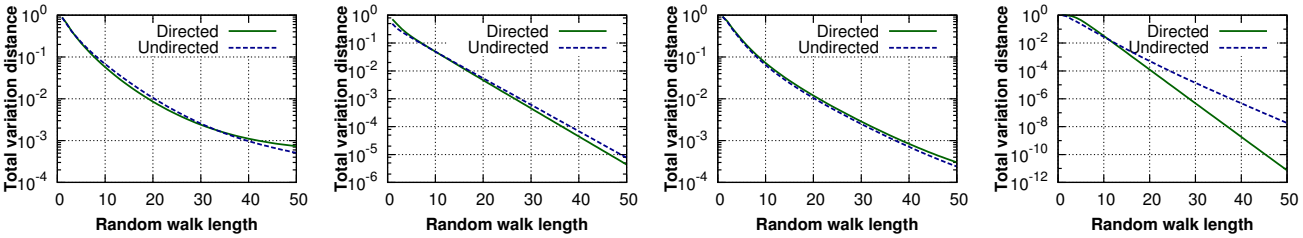
We convert each of the SCC's computed above from the directed graph form to an undirected graph by completing the in- and out-degrees, and omit loops to obtain simple graphs. The resulting graphs and their statistics are shown in Table 3. While the same graph size is maintained, the difference in the number of edges and graph density between both forms of the graph is great, and ranges from as low as 11% in Slashdot (agreeing with previous results on other datasets such as Livejournal experimented with in [23]) to as high as 50% as in Gnutella and Wiki-vote.

6.2.2 Results and Discussion

Now we proceed to measure the mixing time for both directed and undirected graphs. Same as in section 3, we use the definition of the mixing time, with the stationary distribution properly computed for the different graphs based on their types. For each graph, we measure ϵ—the distance between the stationary distribution and the accumulated distribution after t steps. We repeat this process for each graph, and by beginning from 1000 different nodes as sources of initial distributions in order to capture the pattern of mixing.

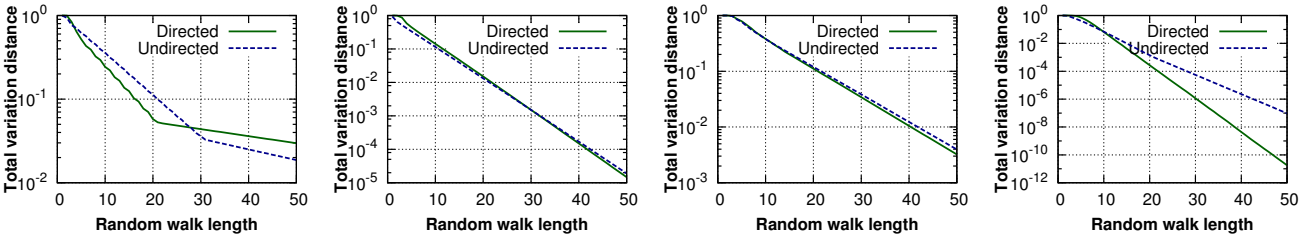
The results of the measurements are shown in Figure 6 and Figure 7. Figure 6 shows a comparison between the average ϵ among 1000 sources for varying t—from 1 to 100, whereas Figure 7 shows maximum ϵ for a given t among the 1000 sources, making t the mixing time for that ϵ by definition. Contrary to what is anticipated due to the difference in the graph structure introduced by omitting edge direction, we find that the average mixing time does change in most cases, as shown in figures 6(a) through 6(c), which are representative social graphs. More interesting, we find that the mixing time of the directed graph is even better than that of the undirected graph, as it is the case in Figure 6(d).

While the mean computed over all ϵ's for a given t is meaningful for the average node, it does not capture the worst case scenario which is of interest in many cases. For example, for an anonymous communication systems suggested on top of social networks, it is always better to prove guarantees with lower-bounds. Lower bound guarantees are satisfied by the mixing time in the definition, and satisfied for the maximum ϵ among all distributions for a given t. For the same experiment above, we plot maximum ϵ for different t, where the results are shown in Figure 7. As the per the definition of the mixing time, no difference between measurements for both types of graphs is observed (in Figure 7(b) and Figure 7(c)). Furthermore, the same advantage of mixing time pointed out earlier for Gnutella dataset is also translated into a better mixing time when considering the maximum among all sources, as shown in Figure 7(d). Finally, while the average mixing characteristics of Epinion shown in Figure 6(a) are tied for both directed and undirected graphs, the maximum ϵ computed among all sources for a given t differs slightly, as shown in Figure 7(a). Furthermore, in the same figure, we observe that a steep region of the convergence to the stationary distribution ends at some point, which happens earlier in the directed graph case. Even before switching the convergence rate, both graphs provide similar mixing characteristics, though the directed graph mixes better in the active region (for $t < 20$).



(a) Epinion (b) Wiki-vote (c) Slashdot (d) Gnutella

Figure 6: A measurement of the mixing time (average case) of undirected graphs before and after omitting directions.



(a) Epinion (b) Wiki-vote (c) Slashdot (d) Gnutella

Figure 7: A measurement of the mixing time (worst case, the maximum ϵ for each walk length) of undirected graphs and after converting them into the directed form by completing their in- and out-degree.

7. THE IMPACT OF LINK DYNAMICS

All Sybil defenses that use social networks did not consider social churn as part of their design and analysis. This social churn happens for nodes and edges. Node churn is due to appearance and disappearance of nodes (by joining or leaving the network) while edge churn is due to the creation or removal of edges between nodes in the network. As we have shown in section 2.4, there are many efforts to understand the dynamics of social networks and the way they affect topological structures, but none considered understanding how much these dynamics affect the mixing time.

Here, we consider measuring the mixing time in dynamic graphs. We define the mixing time, explain our method for deriving datasets for which the mixing time is meaningful and well defined, and show results of measuring the mixing time with insights on the applications that use the quality of mixing time as a metric for their operation performance.

7.1 Defining Mixing Time for Dynamic Graphs

In static graphs, the mixing time captures the length of the random walk, at worst, required for reaching a constant distance from the stationary distribution. Extending this definition to dynamic graphs is quite easy. Given that dynamic graphs are multiple snapshots of the "same" set of nodes, computing the mixing time for the dynamic graph would add one additional dimension to the problem, which is the time.

Let the dynamic graph $G = \{G_1, G_2, \ldots, G_g\}$ where $G_j (1 < j \leq g)$ is the graph at time j. Extending the mixing time definition in (3) to cover the dynamic graph produces

$$T(\epsilon) = \max_j \max_i \min\{t : |\pi_j - \pi_j^{(i)} \mathbf{P}_j^t|_1 < \epsilon\}, \qquad (4)$$

where $\mathbf{P}_j$ is the transition matrix of G_j and π_j is a stationary distribution computed from $\mathbf{P}_j$. $\pi_j^{(i)}$ is the starting distribution at v_i at time j. π_j depends on graph type, undirected or directed, and can be computed as in section 3 and section 6, respectively.

The intuition of the definition in (4) is simple. While the mixing time in a static graph, by definition, provides an upper bound on t from all sources by accounting for the worst mixing source, the definition in (4) adds the time as another dimension in the upper bound, so as every source at *any time* would have at least as good as what this bound provides.

Social churn by both node and edge dynamics would enforce constraints on the definition above. By definition, every snapshot is supposed to have the same set of nodes to avoid the impact of changing the graph size on the measured mixing time. In reality, two churn cases violate this definition: (1) nodes in the past but not in the current or future, and (2) nodes in the future or present but not in the past.

On one hand, we need to relax the definition to capture both cases. On the other hand, we need to limit the data so as the mixing time is computed for a set of nodes who are "mostly" associated with each other over time under link dynamics only, but not both edge and node dynamic. Here we emphasize "mostly" since there is no guarantee to have a connected social graph for which the mixing time is well-defined for all nodes at all times. In the following, we use the same defini-

tion for measuring the mixing time in both cases. We elaborate on data collection and cleaning so as to obtain representative graphs for both cases.

7.2 Datasets and Data Preprocessing

Our sources of data are two datasets, the DBLP Computer Science co-authorship graph [24] and a Facebook interaction social graph [47]. Both graphs are available online.

The DBLP Dataset: The original DBLP dataset consists of 943,316 nodes representing authors in computer science and 6,379,554 edges between them, for publication records from March 1936 to May 2011. An edge between two nodes indicates that both nodes co-authored a paper together. The graph consists of 40,685 disconnected components and all edges are between 892,565 nodes. After removing isolated nodes and disconnected components, the remaining graph consists of 769,642 and 3,051,127 undirected edges.

To generate dynamic graphs from the largest connected component of the DBLP graph, we limit ourselves to the period of 2006 to 2010 inclusive. We select each author who has publications at each and every of these years. The result is a multigraph where two nodes would have an edge if they co-authored a paper or more in a given year. Multiple edges could be created between two authors if they co-authored over multiple years. Multiple edges are labeled with respect to the year of publication. The final multigraph has 46,994 nodes and 458,736 edges. We decompose each multigraph to multiple-graphs with respect to the edge label. Finally, as some nodes who published in the given period could be isolated in a certain year, we remove these nodes so as each resulting graph is connected. Statistics of the different resulting graphs are shown in the upper part of Table 5.

To maintain a consistent association of nodes in each graph from the beginning to the end of the evolution for most nodes, and thus have a better estimate for the mixing time for (almost) the same set of nodes under edge dynamics only, we perform a simple type of matching. To do so, we fix a reference graph—the first or last, depending on which has least number of nodes. For each graph other than the reference, we compute the set intersection of nodes in that graph and those in the reference graph. Then, for the same graph, we consider edges that exist only between the resulting nodes in the set intersection and prune all disconnected nodes. By fixing the last graph in the upper part of Table 5 as a reference, we construct graphs under edge dynamics, for which statistics are in the lower part of Table 5. In the rest of this paper, we refer to the first type of graphs as graphs with "different size" and to the second type as graphs with "same size".

The Facebook Dataset: The Facebook dataset [47] is for wall posts in New Orleans regional network, and spans the period from 2004 to 2009. A link between two nodes indicates that the first node has interacted with the second node. Further details on statistics of the entire dataset is in [47]. The volume of interaction begins slow and increases as the time goes. To obtain a dynamic graph from this dataset, we limit ourselves to the last 30 months of interactions. Because the dataset itself is small and does not guarantee that interactions between the same set of nodes persist over the period of time of interest, we consider all interactions in the period of the 30 months and consider each graph snapshot over a period of 6 months. The resulting five graphs are shown in the upper part of Table 4. Notice the high variability in graph size, which is explained by the growth of Facebook in that period [47]. To maintain a consistent association of nodes in each graph, we use the same method explained above. However, we use the first graph as a reference. The resulting graphs and their statistics are in the lower part of Table 4.

7.3 Results and Discussions

First of all, for both types of dynamic graphs and for both datasets we measure basic structural statistics and properties shown in Table 4 and Table 5. These statistics are as follows

- *Graph size:* number of nodes and number of edges.
- *Assortativity* (of degree; also known as the Pearson correlation coefficient), is a value in [-1, 1] which indicates the correlation between similar degree nodes.
- *Transitivity:* (value in [0, 1]) fraction of all possible triangles (including triads, two edges with a shared vertex between them) which are in fact triangles.
- *Clustering coefficient:* is the average (thus in [0, 1]) of local clustering coefficient for all nodes. The local clustering coefficient for a node is the fraction of possible triangles that go through that node.
- *Diameter:* the longest of eccentricities among all nodes in the graph. The eccentricity of a node is the longest shortest path from it to other nodes in the graph.
- *Radius:* shortest eccentricity.

7.3.1 Basic Dynamic Graph's Characteristics

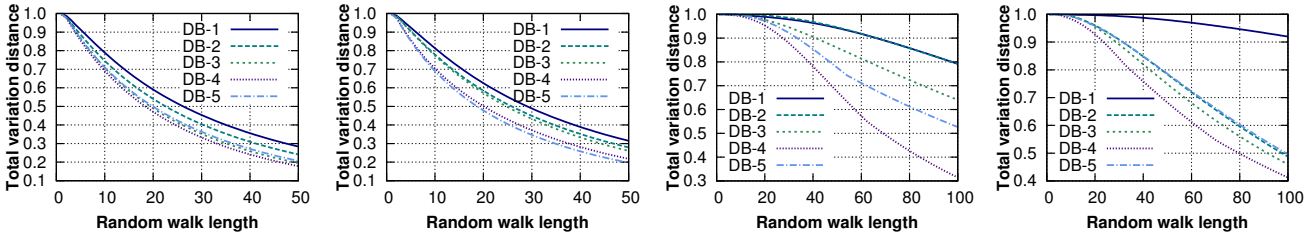
The measurements of these basic characteristics are shown in Table 4 and Table 5. While the general tendency is that graphs in the DBLP dataset tend to have similar size—due to the way these graphs are created—slight changes in the topological characteristics can be observed, and are explained by a decrease in the clustering coefficient and variability in the diameter and radius as the time goes (upper part of Table 5). Furthermore, when using the preprocessing method explained above for deriving a consistent graph for its past and future states, we observe that while the size of the graph

Table 4: Statistics of Facebook time-varying graphs. Metrics of comparison are number of nodes (n), number of edges (m), assortativity (ga), transitivity (gt), average clustering coefficient (gc), diameter (gd), and radius (gr).

Graph	Facebook (different size graphs)						
	n	m	ga	gt	gc	gd	gr
FB-1	9154	23245	0.185	0.087	0.102	19	10
FB-2	13288	37908	0.198	0.085	0.101	18	10
FB-3	16540	42427	0.124	0.075	0.092	19	10
FB-4	23879	59190	0.136	0.068	0.085	21	11
FB-5	35665	86525	0.144	0.067	0.084	18	10
Graph	Facebook (same size graphs)						
	n	m	ga	gt	gc	gd	gr
FB-1	9154	23245	0.185	0.087	0.102	19	10
FB-2	7897	24646	0.221	0.094	0.116	13	8
FB-3	7389	19895	0.124	0.079	0.010	14	8
FB-4	7089	18117	0.140	0.076	0.088	16	9
FB-5	6452	13461	0.088	0.068	0.079	18	10

Table 5: Statistics of DBLP time-varying graphs. Metrics of comparison are number of nodes (n), number of edges (m), assortativity (ga), transitivity (gt), average clustering coefficient (gc), diameter (gd), and radius (gr).

Graph	DBLP (different size graphs)						
	n	m	ga	gt	gc	gd	gr
DB-1	31704	71994	0.262	0.338	0.483	26	14
DB-2	33012	79475	0.230	0.325	0.480	27	14
DB-3	33923	84125	0.281	0.331	0.467	24	13
DB-4	33071	82282	0.225	0.297	0.453	23	12
DB-5	26150	62161	0.186	0.249	0.419	24	13
Graph	DBLP (same size graphs)						
	n	m	ga	gt	gc	gd	gr
DB-1	18984	42478	0.206	0.303	0.447	22	12
DB-2	20152	48515	0.210	0.304	0.447	27	14
DB-3	21273	53873	0.245	0.307	0.436	22	12
DB-4	21472	55731	0.198	0.269	0.424	22	12
DB-5	26150	62161	0.186	0.249	0.419	24	13



(a) Mean (same) (b) Mean (diff.) (c) Max (same) (d) Max (diff)

Figure 8: A measurement of the mixing time—worst case, maximum ϵ for each walk length vs. the mean—for dynamic graph generated from the DBLP dataset using the two methods for same and different sizes. [Best viewed in colors]

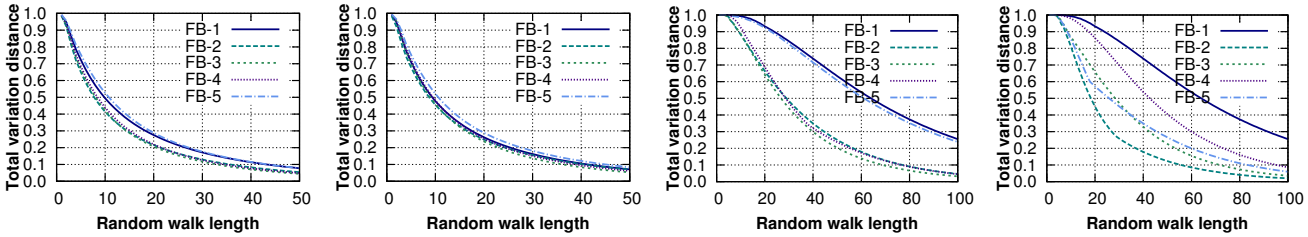
is affected (which is natural) other properties such as the clustering coefficient, diameter, and radius are minimally affected. This is indeed surprising, as the size of the different graphs is reduced greatly from its original size. For example, while the size of DB-4 decreases by about 35%, the radius and diameter are still almost the same. Other properties are affected in the same way (or even less) for similar change in graph size across the different snapshots.

Similar observations are made on the Facebook dataset in Table 4. Notice that the first set of graphs (with different size) indeed differs greatly in size due to node dynamics. Despite that, this change in size is minimally reflected on the radius and diameter while it is more significant for the clustering coefficient. Also, as the time goes, the general tendency in these graphs is that their transitivity and assortativity decrease despite size growth (in general). When using the above method for generating graphs with similar size, we observe that (in general) the radius and diameter (as well as other properties) decrease more significantly than in DBLP.

7.3.2 *The Mixing Time of Dynamic Graphs*

Here we describe the results of measuring the mixing time in dynamic graphs. We use the same settings (number of initial distributions and length of walks) as in section 3. The results are shown in Figure 8 (for max and mean ϵ measurements of DBLP) and Figure 9 (for Facebook). On these plots, we make several quick remarks.

Most importantly, and as anticipated, the mixing time is time-dependent. Both mean and max ϵ change significantly as the graph evolve over time, regardless to what considerations are made for node associations within the graph. This distance is as high as about 0.1 in the DBLP dataset for the average ϵ as shown shown in Figure 8(a) and as high as 0.12 for graphs with the different size as shown in Figure 8(b), for $t = 20$. Also, the difference ϵ is as high as 0.5 for same size graphs (shown in Figure 8(c)) and 0.52 in graphs with different size (shown in Figure 8(d)) for the maximum ϵ which captures the mixing time by definition, for $t = 100$.



(a) Mean (same) (b) Mean (diff.) (c) Max (same) (d) Max (diff)

Figure 9: A measurement of the mixing time—worst case, maximum ϵ for each walk length, vs. the mean—for dynamic graph generated from Facebook dataset using the two methods for same and different sizes. [Best viewed in colors]

Similar observations are made on the Facebook graphs, where results are in Figure 9. The dynamic graph of Facebook exhibits less variability in the mixing pattern at average than in DBLP (in Figure 9(a) and Figure 9(b)) but more on average in graphs with same size than the raw graphs with different size. Variability of the worst mixing, which captures the mixing of the poorest source, is more in graphs with different size than with the same size, though both are quite variable with respect to graphs in the same category (in Figure 9(d) and Figure 9(d). The difference among different datasets with our previous measurement in [33] and the difference in the same dataset in this way agrees with the measurements on the volume-wise evolution of the Facebook dataset in [47], where it has been shown that a growth in the communication volume (the pattern in graphs with different size), which decays as the edge between nodes ages (the pattern in graphs with same size). This latter part explains the decreased convergence rate in graphs with same size due to their distortion over time. For more details, see [32].

8. CONCLUSION

In this work, we extended upon our previous results of measuring the mixing time and advanced this vein of research in several directions. First, we have shown that the mixing time of social graphs is related to their degeneracy, which captures cohesiveness of graphs. We use this finding to explore methods for improving the mixing time of slow mixing social graphs. In parallel to that, we have investigated the impact of link direction on the mixing characteristics and have shown that, counterintuitively, directed graphs are mostly faster mixing than undirected graphs. Last, we defined and measured the mixing time of dynamic graphs and we found that the mixing time of these graphs is time-varying, as anticipated. Under certain circumstances, the mixing quality of these graphs decay as the time goes, agreeing with other findings in the literature on these graphs.

Acknowledgement— A preliminary version of this work appeared in [31] as an invited contribution. Appendix B and Appendix C are intentionally omitted and are available upon request. They are not cited in any place through this work. They are parallel contributions to the material presented in this work.

This research was supported by NSF grant CNS-0917154 and a generous research grant from KAIST.

9. REFERENCES

[1] S. Arora, E. Hazan, and S. Kale. $O(\sqrt{\log n})$ approximation to SPARSEST CUT in O (n2) time. In *FOCS*, pages 238–247, 2004.

[2] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC*, pages 222–231, 2004.

[3] K. Avrachenkov, B. F. Ribeiro, and D. F. Towsley. Improving random walk estimation accuracy with uniform restarts. In *WAW*, 2010.

[4] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD*, pages 44–54. ACM, 2006.

[5] V. Batagelj and M. Zaversnik. An $O(m)$ algorithm for cores decomposition of networks. Arxiv preprint cs/0310049, 2003.

[6] E. M. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *MobiHoc*, 2007.

[7] G. Danezis, C. Lesniewski-Laas, M. Kaashoek, and R. Anderson. Sybil-resistant DHT routing. *ESORICS*, pages 305–318, 2005.

[8] G. Danezis and P. Mittal. SybilInfer: Detecting sybil nodes using social networks. In *NDSS*, 2009.

[9] M. Dellamico, , and Y. Roudier. A measurement of mixing time in social networks. In *IWSTM*, 2009.

[10] P. Erdős and A. Hajnal. On chromatic number of graphs and set-systems. *Acta Mathematica Hungarica*, 17(1):61–99, 1966.

[11] M. Gjoka, C. T. Butts, M. Kurant, and A. Markopoulou. Multigraph sampling of online social networks. *JSAC*, 2011.

[12] J. Golbeck. The dynamics of web-based social networks: Membership, relationships, and change. *First Monday*, 12(11), 2007.

[13] T. Isdal, M. Piatek, A. Krishnamurthy, and

T. Anderson. Privacy preserving p2p data sharing with oneswarm. In *SIGCOMM*, 2010.

[14] M. Jerrum and A. Sinclair. Conductance and the rapid mixing property for markov chains: the approximation of the permanent resolved. In *STOC*, pages 235–244. ACM, 1988.

[15] S. Kirkland. Fastest expected time to mixing for a Markov chain on a directed graph. *Linear Algebra and its Applications*, 2010.

[16] J. Kleinberg. Cascading behavior in networks: Algorithmic and economic issues. *Algorithmic game theory*, pages 613–32, 2007.

[17] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. *Link Mining: Models, Algorithms, and Applications*, pages 337–357, 2010.

[18] M. Kurant, M. Gjoka, C. T. Butts, and A. Markopoulou. Walking on a Graph with a Magnifying Glass. In *ACM SIGMETRICS '11*, 2011.

[19] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Predicting positive and negative links in online social networks. In *WWW*, 2010.

[20] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: dens. laws, shrinking dim. and possible explanations. In *KDD*, 2005.

[21] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM TKDE*, March 2007.

[22] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *CoRR*, abs/0810.1355, 2008.

[23] C. Lesniewski-Lass and M. F. Kaashoek. Whānau: A sybil-proof distributed hash table. In *NSDI*, pages 3–17, 2010.

[24] M. Ley. The DBLP computer science bibliography: Evolution, research issues, perspectives. In *SPIR*, 2009.

[25] D. Lick and A. White. k-Degenerate graphs. *Canadian J. of Mathematics*, 22:1082–1096, 1970.

[26] S. Marti, P. Ganesan, and H. Garcia-Molina. DHT routing using social links. *Peer-to-Peer Systems III*, pages 100–111, 2005.

[27] A. Mashhadi, S. Mokhtar, and L. Capra. Habit: Leveraging human mobility and social network for efficient content dissemination in manets. In *WoWMoM*, 2009.

[28] A. Mislove, M. Marcon, P. K. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *Internet Measurement Comference*, pages 29–42, 2007.

[29] A. Mislove, A. Post, P. Druschel, and K. Gummadi. Ostra: Leveraging trust to thwart unwanted communication. In *NSDI*, 2008.

[30] A. Mohaisen, N. Hopper, and Y. Kim. Incorporating trust into social network-based sybil defenses. In *INFOCOM*, 2011.

[31] A. Mohaisen, H. Tran, N. Hopper, and Y. Kim. Understanding social network properties for trustworthy computing. In *SIMPLEX*, 2011.

[32] A. Mohaisen, H. Tran, N. Hopper, and Y. Kim. Understanding the mixing patterns of social networks. Technical report, UMN, 2011.

[33] A. Mohaisen, A. Yun, and Y. Kim. Measuring the mixing time of social graphs. In *IMC*, pages 383–389. ACM, 2010.

[34] S. Nagaraja. Anonymity in the wild. In *PETS*, 2007.

[35] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Stanford InfoLab, 1998.

[36] D. Quercia and S. Hailes. Sybil attacks against mobile users: friends and foes to the rescue. In *INFOCOM*, 2010.

[37] B. F. Ribeiro and D. F. Towsley. Estimating and sampling graphs with multidimensional random walks. In *IMC*, 2010.

[38] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *ISWC*, LNCS, pages 351–368. Springer, 2003.

[39] A. Sinclair. Improved bounds for mixing rates of mc and multicommodity flow. *Comb., Prob. & Comp.*, 1:351–370, 1992.

[40] M. Sirivianos, K. Kim, and X. Yang. Introducing Social Trust to Collaborative Spam Mitigation. INFOCOM, 2011.

[41] J. Sun, C. Faloutsos, S. Papadimitriou, and P. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *KDD*, 2007.

[42] L. Tang, H. Liu, J. Zhang, and Z. Nazeri. Community evolution in dynamic multi-mode networks. In *KDD*, 2008.

[43] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community ident. in dynamic social networks. In *KDD*, 2007.

[44] H. Tong, S. Papadimitriou, J. Sun, P. Yu, and C. Faloutsos. Colibri: fast mining of large static and dynamic graphs. In *KDD*, 2008.

[45] N. Tran, J. Li, L. Subramanian, and S. S. Chow. Optimal sybil-resilient node admission control. In *INFOCOM*, 2011.

[46] N. Tran, B. Min, J. Li, and L. Subramanian. Sybil-resilient online content voting. In *USENIX NSDI*, 2009.

[47] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *WOSN*, 2009.

[48] B. Viswanath, A. Post, K. P. Gummadi, and A. Mislove. An analysis of social network-based

sybil defenses. In *SIGCOMM*, 2010.
[49] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, and B. Y. Zhao. User interactions in social nets and their implications. In *EuroSys*, 2009.
[50] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao. SybilLimit: A near-optimal social net defense against sybil attacks. In *S&P*, 2008.
[51] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: defending against sybil attacks via social networks. In *SIGCOMM*, pages 267–278, 2006.

APPENDIX

A. BOUNDING THE ERROR IN ESTIMATION π FOR DIRECTED GRAPHS

THEOREM 1. *Let* $|\pi_t^{(i)} - \pi|_1 \leq \epsilon$ *and* $|\pi_t^{(i)} - \pi_\ell^{(*)}|_1 \leq \epsilon'$, *then* $\epsilon - \epsilon' \leq \delta$ *where* $\delta \geq |\pi_\ell^{(*)} - \pi|_1$ *and* $\epsilon \leq 2\max(\epsilon', \delta)$. *Furthermore, for large enough* ℓ, $\epsilon \approx \epsilon'$.

PROOF. The proof of all statements follows from the total variation property and the triangular inequality. Recall that the ultimate stationary distribution for a directed graph is $\pi = J\mathbf{P}^t$, where $t \to \infty$, $\mathbf{P}$ is the transition probability matrix as defined in section 6.1.1, and J is a $(1 \times n)$-vector in which all entries are ones. Using the method used for estimating the stationary distribution π in section 6.1.2, we recall the following quantities:

- π is the ideal stationary distribution of every random walk defined on the graph $\vec{G}$ according to transitions defined by $\mathbf{P}$ as in above.
- $\pi_\ell^{(*)}$ is an estimate of π computed according to the method in section 6.1.2; i.e., $\pi_\ell^{(*)} = \pi^{(*)}\mathbf{P}^\ell$.
- $\pi_t^{(i)}$ is the distribution of a random walk after t steps by beginning from a source v_i; i.e., $\pi_t^{(i)} = \pi^{(i)}\mathbf{P}^t$.

Now, we want to bound ϵ, the statistical distance (total variation distance) between the (ideal) stationary distribution and the distribution of a random walk on the graph after t steps beginning from node v_i (which could be any node). Without losing generality, let the following be true (Both (5) and (6) are given in the statement of the theorem)

$$|\pi_\ell^{(*)} - \pi|_1 \leq \delta \tag{5}$$
$$|\pi_\ell^{(*)} - \pi_t^{(i)}|_1 \leq \epsilon' \tag{6}$$

We are here interested in bounding $|\pi_t^{(i)} - \pi|_1$. Using the triangular inequality, and from (5) and (6), we have

$$\begin{aligned} |\pi_t^{(i)} - \pi|_1 &\leq |\pi_\ell^{(*)} - \pi|_1 + |\pi_\ell^{(*)} - \pi_t^{(i)}|_1 \\ &\leq \delta + \epsilon' \\ &\leq 2\max(\delta, \epsilon') \\ &\geq \epsilon \end{aligned} \tag{7}$$

However, since δ is a function of ℓ that goes to 0 as $\ell \to \infty$, and ℓ in our experiments is set to be large, particularly $\ell \gg t$ from which we obtain $\epsilon' \gg \delta$, the last inequality can be further relaxed to $\epsilon \approx \epsilon'$. □

Notice that δ and ϵ' are distances computed from a fixed distribution to two different stationary distributions, thus formally speaking this argument wouldn't hold in general. However, because of the graph structure which is fast mixing by nature and the length of the random walk that we use for both cases, it is mild to show correctness under this assumption for a relatively small t (say, 100). In experiments, we use ℓ to be 100 times t so that the argument holds.

B. IMPACT OF IMPROVED MIXING TIME ON THE PERFORMANCE OF SYBIL DEFENSES

The contents of this section have been intentionally omitted. An extended revision including these sections will be posted soon, and is available upon request

C. SIZE PATTERN OF MULTI-CORES IN SLOW MIXING GRAPHS

The contents of this section have been intentionally omitted. An extended revision including these sections will be posted soon, and is available upon request