2014 IEEE International Conference on Data Mining

Locally Estimating Core Numbers

Michael P. O'Brien, Blair D. Sullivan
Department of Computer Science
North Carolina State University
Raleigh, North Carolina, 27607
Email: {mpobrie3,blair_sullivan}@ncsu.edu

***Abstract*—Graphs are a powerful way to model interactions and relationships in data from a wide variety of application domains. In this setting, entities represented by vertices at the "center" of the graph are often more important than those associated with vertices on the "fringes". For example, central nodes tend to be more critical in the spread of information or disease and play an important role in clustering/community formation. Identifying such "core" vertices has recently received additional attention in the context of *network experiments*, which analyze the response when a random subset of vertices are exposed to a treatment (e.g. inoculation, free product samples, etc). Specifically, the likelihood of having many central vertices in any exposure subset can have a significant impact on the experiment.**

We focus on using *k-cores and core numbers* to measure the extent to which a vertex is central in a graph. Existing algorithms for computing the core number of a vertex require the entire graph as input, an unrealistic scenario in many real world applications. Moreover, in the context of network experiments, the subgraph induced by the treated vertices is only known in a probabilistic sense. We introduce a new method for estimating the core number based only on the properties of the graph within a region of radius δ around the vertex, and prove an asymptotic error bound of our estimator on random graphs. Further, we empirically validate the accuracy of our estimator for small values of δ on a representative corpus of real data sets. Finally, we evaluate the impact of improved local estimation on an open problem in network experimentation posed by Ugander et al.

I. Introduction

In a graph modeling complex interactions between data instances, the connectivity of the vertices often yields useful insights about the properties of the represented entities. For example, in a social network, it is important to distinguish between a person who is a member of a relatively large, tight-knit community, and a person who exists on the periphery without much involvement with any cohesive communities. This type of connectivity is often not well-correlated with the vertex degree (raw number of connections), leading to the introduction of several more sophisticated metrics. Here, we focus on the *core number* [1]. To build intuition, consider the setting of a graph representing friendships in a social network – the vertices with large core numbers form a set where people tend to have many common friends, whereas the subset of vertices with small core numbers form a group where most people do not know one another. The applications of core numbers are numerous, well studied, and permeate a variety of domains, such as community detection [2], [3], virus propagation [4], data pruning [5], and graph visualization [6].

Existing algorithms for computing core numbers run in linear time with respect to the number of vertices and edges of the graph [1], but require the entire graph as input and simultaneously calculate the core number for every vertex. There are a number of scenarios in which current approaches are insufficient. First, one might only be concerned with the properties of a small subset of query vertices[1]. If the query vertices constitute a small fraction of the entire graph, it would be much more time- and memory-efficient to locally estimate the core numbers of those specific vertices rather than using the global algorithm, regardless of the fact that it is linear. Second, the entire graph may be unknown and/or infeasible to obtain due to scale, privacy, or business strategy reasons. For example, suppose one wanted to investigate patterns of phone calls between cell phone users. It would be possible to contact a small set of people and ask them to voluntarily log their calls for one month. However, without access to the telecommunication companies' private records, expanding the domain to the national or global level would not be possible. Finally, current methods for determining core numbers cannot be applied to solve problems in the domain of network experimentation, as we describe below.

A *network treatment experiment* is a randomized experiment in which subjects are divided into two groups: those receiving treatment and those receiving none (or a placebo). Network treatments differ from other randomized experiments in that the effects of the treatment (or lack thereof) on a given subject are assumed to be dependent on the experiences of other subjects in the experiment. Randomly assigning subjects into two groups (treated versus untreated) is equivalent to randomly partitioning the vertices of a graph into two sets. Previous work has given methods to calculate degree probabilities of a randomly partitioned graph [7], but an analogous algorithm for the core numbers is an open problem. Given the results of Kitsak et al. on the importance of core numbers in spreading information [4], an algorithm to predict the likelihood of a given subject having a large core number would help researchers better understand the impact of their experimental design. For example, researchers conducting market testing on a product that relies on social interaction (such as a new social networking site, online game, etc.) would have a greater ability to see whether early access to their product will generate widespread excitement among the test subjects. Additionally, if certain groups of vertices (say, females participating in the experiment) are more likely to be core exposed than the others, we can reduce the bias of the estimate of the treatment effect by upweighting the probability that underrepresented vertices (males) are treated.

[1]For example, vertices which have some common metadata (e.g. age, physical location, etc.) of interest to the user that is not represented explicitly in the graph

All of the challenges mentioned above could be addressed by estimating the core number of a vertex based on local graph properties. The existence of an accurate non-global estimator is intuitively well-grounded, as it has been shown that addition and deletion of edges can only affect the core numbers of a limited subset of vertices [8]. This suggests that in spite of the fact that computing core numbers exactly requires knowledge of the whole graph, the core number of a given vertex may often depend on a much smaller subgraph. Moreover, even if a core number estimate has a small error, it may still be useful in applications. In particular, it is often sufficient to delineate between vertices with a "large" core number and those with a "small" core number. That is to say, while a vertex in the 50-core is substantially more well-connected than one in the 2-core, it may be functionally identical to a vertex in the 51-core in downstream analysis.

This work introduces a new local estimator of the core number at a specified vertex, which allows a user to tune the balance between accuracy and computational complexity by varying the size of the local region around the query vertex that it considers. We prove that in an Erdös-Rényi random graph, the error of our approximation at each vertex asymptotically almost surely grows arbitrarily slowly with the size of the graph. We also empirically evaluate the estimates with respect to the actual core numbers on a representative corpus of real-world graphs of varying sizes. The results on these graphs demonstrate that high accuracy can be achieved even when considering only a small local region. Finally, we show how our estimators can be applied to address the aforementioned open problem in network treatment experiments. Specifically, we give an algorithm to tighten the upper bound on the core number of a vertex given in [7], and evaluate the impact empirically on a sample experiment.

II. Background and Definitions

In this paper, all graphs are simple, undirected, and unweighted. Unless otherwise specified, G denotes a graph with vertex set V and edge set E, where $n = |V|$. The number of edges incident to a vertex v is the degree of v, denoted $d(v)$. We also assume that $\forall v \in V$, $d(v) > 0$ (isolated vertices are not relevant to our algorithms and analysis). The notation $G[S]$ denotes the subgraph of G *induced* on the vertices $S \subseteq V$. In other words, $G[S] = (S, F)$ is the graph with vertex set S and edge set $F = \{(u, v) \in E | \, u, v \in S\}$. Given a function f, we say a set of vertices $u_1, u_2, \ldots$ is *f-ordered* if for all i, $f(u_i) \leq f(u_{i+1})$.

To quantify the idea of a "central" vertex, we formalize the notion of being in a well-connected subgraph:

Definition 1 ([1]): The *k-core* of G, denoted C_k, is the maximal induced subgraph of G with minimum degree at least k.

Clearly, a graph with minimum degree at least k also has minimum degree at least $k - i$ for $i = 1, 2, \ldots$, so $C_k \subseteq C_{k-1}$. Thus, for sake of specificity, we measure the degree to which a vertex is central by the deepest core in which it participates:

Definition 2: The *core number* of v, denoted $k(v)$, is the largest $k \geq 0$ such that $v \in C_k$.

The term *core structure* will be used broadly to describe all properties of or relating to the k-cores of G. A common global metric measures the depth of this structure:

Definition 3: The *degeneracy* of G is the largest k for which $|C_k| > 0$; a graph with degeneracy at least D is said to be *D-degenerate*.

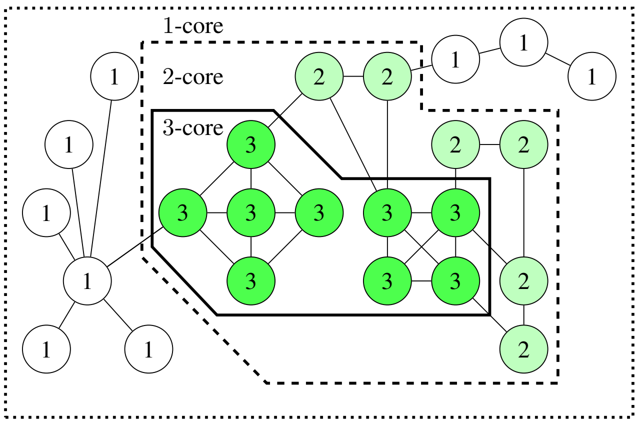


Fig. 1: Sample graph with its k-cores and core numbers labeled.

The core number of a vertex v in a graph G can be found by performing Algorithm 1 (from [1]), which finds a *core decomposition* of G. Beginning with $i = 0$, the algorithm deletes vertices with degree at most i (and their incident edges) until there are no such vertices remaining. The removal of edges incident to a vertex may cause its neighbors whose degrees were initially larger than i to have their degree reduced to at most i. In this case, those neighbors would be also be deleted in the degree i phase. Once all vertices remaining in G have $d(v) > i$, i is incremented by 1 and the process repeats until G no longer has any vertices. The core number of a vertex is the value of i when it is removed from the graph and the degeneracy of G is the value of i when the last vertex is removed. Since each vertex and edge is removed exactly once, a core decomposition can be completed in $O(|V| + |E|)$ time.

Algorithm 1 Core decomposition

```
   INPUT: Graph $G = (V, E)$
   OUTPUT: $k(v)\ \forall v \in V$
1: $i \leftarrow 0$
2: while $|V| > 0$ do
3:     while $\exists v : d(v) \leq i$ do
4:         $k(v) \leftarrow i$
5:         $V \leftarrow V \backslash \{v\}$
6:         $E \leftarrow E \backslash \{(u, v) | u \in V\}$
7:     end while
8:     $i \leftarrow i + 1$
9: end while
```

The basic k-core decomposition has been tailored to meet additional constraints. Montressor et al. [9] and Jakma et al. [10] proposed methods by which the core decomposition could be computed in parallel. Li et al. [8] and, later, Saríyüce et al. [11] described ways to update the core numbers of vertices in a dynamic graph without recomputing the full core decomposition each time a vertex or edge is added. Finally, Cheng et al. [12] gave an alternate implementation of the core decomposition for systems with insufficient memory to store the entire graph at once.

III. Local Estimation and Theory

In order to estimate core numbers efficiently without knowledge of the entire graph, we will restrict the domain of our algorithms to a localized subset around a vertex:

Definition 4: The *δ-neighborhood* of a vertex v, denoted $N_\delta(v)$, is the set of vertices at distance [2] at most δ from v.

The estimation algorithms will vary the size of their input by allowing δ to range from zero to Δ, the *diameter* of G (the maximum distance among all pairs of vertices).

A. Neighborhood-based estimation

A relatively naïve approach to local estimation would be to compute a core decomposition of the subgraph of G induced on the δ-neighborhood of v and use the resulting core number of v as the estimate:

Definition 5: Let the *induced estimator*, $\breve{k}_\delta(v)$, be the core number of v in $G[N_\delta(v)]$.

By increasing δ, the induced subgraph captures a progressively larger fraction of the graph, improving the estimate until $\breve{k}_\delta(v) = k(v)$ (which is guaranteed to happen at $\delta = \Delta$, but could happen for significantly smaller δ). Note that when $\delta = 1$, the estimator will only be close to the core number if the neighbors of v are highly interconnected (so there is a subtle relationship to clustering coefficient).

Lemma 1: Let $G = (V, E)$ be a graph. For all $v \in V$, $0 = \breve{k}_0(v) \le \breve{k}_1(v) \le \breve{k}_2(v) \le \cdots \le \breve{k}_\Delta(v) = k(v)$

Proof: First we will establish the boundary conditions on our inequality. Because $N_\Delta(v) = G$, $\breve{k}_\Delta(v) = k(v)$. Since $N_0(v) = \{v\}$, $\breve{k}_0 = 0$. Additionally, for all $\delta \in [1, \Delta]$, $N_{\delta-1}(v) \subseteq N_\delta(v)$. This implies that for all $u \in V$, the degree of u in the subgraph induced by the $(\delta-1)$-neighborhood of v can be no greater than the degree of u in the δ-neighborhood of v. Thus v cannot participate in a deeper core with respect to the $(\delta-1)$-neighborhood than it does with respect to the δ-neighborhood, which makes $\breve{k}_{\delta-1} \le \breve{k}_\delta$. ■

In order to make a more sophisticated estimation, let us consider the information gained as δ increases. At $\delta = 0$, we assume that $d(v)$ is known. Because the k-core requires all vertices to have minimum degree k, $k(v)$ can not be greater than $d(v)$. Thus, $d(v)$ itself can be an estimate of $k(v)$. Expanding out to $\delta = 1$ allows information about v's immediate neighbors to be utilized. Suppose the core numbers of the neighbors of v were known. It would then be possible to compute $k(v)$ precisely using the following two lemmas (previously shown by Montresor et al):

Lemma 2 ([9]): A vertex v is in the j-core of a graph G if and only if v has at least j neighbors in the j-core.

We now give a closed-form algebraic expression for the largest j satisfying Lemma 2.

[2]We use the typical shortest-path distance function throughout

Lemma 3 ([9]): Let $u_1, u_2, \ldots$ be the k-ordered neighbors of v. Then

$$k(v) = \max_{1 \le i \le d(v)} \left(\min\left(k(u_i), d(v) - i + 1\right)\right).$$

Proof: By the definition of core number and Lemma 2, $k(v)$ is the largest j in $0 \le j \le d(v)$ so that v has at least j neighbors with core number at least j. For each u_i ($1 \le i \le d(v)$), v has $d(v) - i + 1$ neighbors with core numbers at least $k(u_i)$, since $k(u_i) \le k(u_{i+1})$. If $k(u_i) \le d(v) - i + 1$, v has at least $k(u_i)$ neighbors in the $k(u_i)$-core. Otherwise, v has at least $d(v) - i + 1$ neighbors in the $(d(v) - i + 1)$-core. This shows that the core number of v must be at least the minimum of $k(u_i)$ and $d(v) - i + 1$ for every i (and thus the maximum over i). Equality follows easily by contradiction. ■

Note that $k(v)$ is the maximum value of the minimum of two functions of i: $k(u_i)$ and $d(v) - i + 1$. With respect to i, $k(u_i)$ is monotonically non-decreasing and $d(v) - i + 1$ is monotonically decreasing. From a geometric perspective, then, the maximum of their minimums occurs at the intersection of the curves $k(u_i)$ and $d(v) - i + 1$ (as stylized in Figure 2).

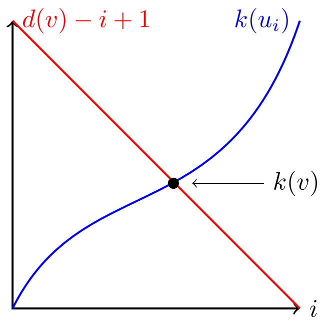


Fig. 2: $k(v)$ is the y-value at intersection of two functions.

Although the core numbers of the neighbors of a vertex may not be known *a priori*, the reasoning behind Lemma 3 gives useful insight into the behavior of $k(v)$. As shown by Cheng et al. [12], an upper bound on $k(v)$ can be achieved if an upper bound on $k(u)$ is known for all $u \in N_1(v)$.

Theorem 1 ([12]): Let $G = (V, E)$ be a graph, $v \in V$, and ψ any function satisfying $\psi(u) \ge k(u) \forall u \in V$. Let v's neighbors be ψ-ordered. Then

$$k(v) \le \max_{1 \le i \le d(v)} \left(\min\left(\psi(u_i), d(v) - i + 1\right)\right).$$

Proof: Substituting $\psi(u_i)$ for $k(u_i)$ in the expression from Lemma 3 can only increase the right hand side, giving an upper bound on $k(v)$. ■

We base our second estimator on the idea of incorporating iterative upper bounds on the core numbers of a vertex's neighbors:

Definition 6: Let the *propagating estimator*, $\hat{k}_\delta$, be the estimator of $k(v)$ given by the recurrence

$$\hat{k}_\delta(v) = \begin{cases} \max\limits_{1 \le i \le d(v)} \left(\min(\hat{k}_{\delta-1}(u_i), d(v) - i + 1)\right) & \text{if } \delta > 0 \\ d(v) & \text{if } \delta = 0 \end{cases}$$

where $u_1, u_2, \ldots$ are the $\hat{k}_{\delta-1}$-ordered neighbors of v.

Pseudocode for computing $\hat{k}_\delta(v)$ is given in Algorithm 2. Essentially, Algorithm 2 first computes the coarsest upper bound ($\hat{k}_0$) for those vertices at distance at most δ from v. Those estimates are used in conjunction with Theorem 1 to compute a slightly finer upper bound, $\hat{k}_1$, for those vertices at distance at most $\delta - 1$ from v. This process "propagates" inwards towards v until its immediate neighbors have $\hat{k}_{\delta-1}$ values, which are used as the upper bounds in formulating $\hat{k}_\delta(v)$. The computational complexity of finding $\hat{k}_\delta$ is linear in the product of δ and the number of edges in $N_\delta(v)$ (see Theorem 2). Since Algorithm 1 is also linear with respect to the number of edges in the graph, the computational complexity of computing $\hat{k}_\delta$ is comparable to that of $\check{k}_\delta$ for small δ.

Algorithm 2 Algorithm for computing $\hat{k}_\delta(v)$

```
    INPUT: Graph $G$, vertex $v$
    OUTPUT: $\hat{k}_\delta(v)$
 1: if $\delta = 0$ then
 2:     return $d(v)$
 3: else
 4:     for $u \in N_1(v)$ do
 5:         Compute $\hat{k}_{\delta-1}(u)$
 6:     end for
 7:     $\hat{k}_{\delta-1}$-order $N_1(v)$
 8:     $\hat{k}_\delta \leftarrow d(v)$
 9:     for $i \in \{1 \ldots |N_1(v)|\}$ do
10:         $j \leftarrow \min(\hat{k}_{\delta-1}(u_i), d(v) - i + 1)$
11:         if $j > \hat{k}_\delta$ then
12:             $\hat{k}_\delta \leftarrow j$
13:         end if
14:     end for
15:     return $\hat{k}_\delta$
16: end if
```

Theorem 2: For a given vertex u, $\hat{k}_\delta(u)$ can be computed in $O(\delta \cdot |E_\delta(u)|)$ time using Algorithm 2, where $E_\delta(u)$ is the edge set of $G[N_\delta(u)]$.

Proof: Fix $\delta \in \{1, 2, \ldots\}$ and $u \in V$. Assume $\forall u' \in N_1(u)$, $\hat{k}_{\delta-1}(u')$ is known. Since $\hat{k}_\delta$ can only take integer values in the interval $[0, \max_{v \in V} d(v)]$, the sorting (line 7) can be done in $O(d(u))$ time using a bucket sort. Once sorted, each neighbor of u is visited once (line 9) to find the minimum, which can also be done in $O(d(u))$ time. Thus computing $\hat{k}_\delta(u)$ from $\{\hat{k}_{\delta-1}(u') | u' \in N_1(u)\}$ has complexity $O(d(u))$.

Using dynamic programming, we can compute and store $\hat{k}_1(w)$ $\forall w \in N_{\delta-1}(v)$, which in turn can be used to compute $\hat{k}_2(w)$ $\forall w \in N_{\delta-2}(v)$, and so on (through $\delta - 1$ iterations) until we have computed $\hat{k}_\delta(v)$. The jth such iteration requires $\sum_{u \in N_{\delta-j}} O(d(u)) = O(E_{\delta-j}(v))$ operations. Since $N_{\delta-1} \subseteq N_\delta$, the total running time is $O(\delta \cdot |E_\delta|)$. ■

Unlike $\check{k}_\delta(v)$, the estimate $\hat{k}_\delta(v)$ is a decreasing upper bound on $k(v)$ as δ increases:

Theorem 3: $\forall v \in V$, $k(v) \leq \hat{k}_\delta(v) \leq \hat{k}_{\delta-1}(v) \cdots \leq \hat{k}_1(v) \leq \hat{k}_0(v) = d(v)$ for any $\delta \geq 1$.

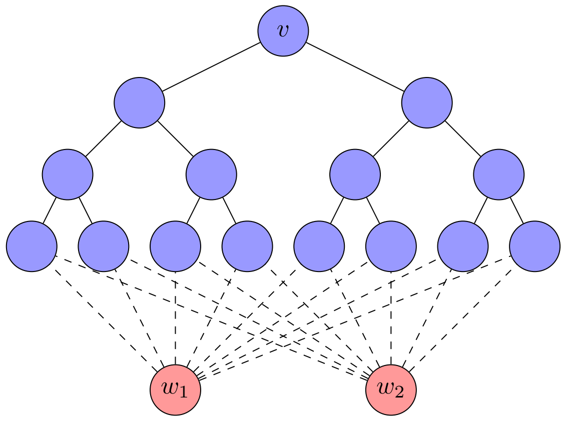


Fig. 3: $T_{2,4}$ is in blue. $T'_{2,4}$ is $T_{2,4}$ plus w_1, w_2 (red), and the dashed edges.

Proof: We first prove that $k(v) \leq \hat{k}_\delta(v)$ for all $\delta \geq 0$ by induction on δ. The base case $k(v) \leq \hat{k}_0(v) = d(v)$ holds, since core number is always bounded by degree. Assume $k(v) \leq \hat{k}_\delta(v)$. Then $\hat{k}_\delta$ is an upper bound ψ as in Theorem 1, and substituting the right hand side with Definition 6, we have $k(v) \leq \hat{k}_{\delta+1}(v)$.

We now prove that $\hat{k}_{j+1}(v) \leq \hat{k}_j(v)$ for all $j \geq 0$ and $\forall v \in V$ by induction on j. Combining Definition 6 with Lemma 3, we see that $\hat{k}_\delta(v)$ is the maximum of the minimum of the functions $\hat{k}_{\delta-1}(u_i)$ and $d(v) - i + 1$ of $1 \leq i \leq d(v)$. Since the maximum of $d(v) - i + 1$ is $d(v)$, $\hat{k}_\delta(v) \leq d(v)$. Because $\hat{k}_0(v) = d(v)$ the base case $\hat{k}_1(v) \leq \hat{k}_0(v)$ is satisfied. Suppose that for some $j \geq 0$, $\hat{k}_j(v) \leq \hat{k}_{j-1}(v)$ for all $v \in V$. By Theorem 1, v has at least $\hat{k}_j(v)$ neighbors that satisfy $\hat{k}_{j-1}(u) \geq \hat{k}_j(v)$ for $u \in N_1(v)$. Each such u also satisfies $\hat{k}_j(u) \leq \hat{k}_{j-1}(u)$, meaning v can have no more than $\hat{k}_j(v)$ neighbors that satisfy $\hat{k}_j(u) \geq \hat{k}_j(v)$. Thus $\hat{k}_{j+1}(v) \leq \hat{k}_j(v)$. ■

B. Structures leading to error

One natural question is whether either $\hat{k}_\delta$ or $\check{k}_\delta$ has bounded error (is a constant-factor approximation of the core number). Unfortunately, there are extremal constructions forcing unbounded error for both estimators; both are based on $T_{j,\ell}$, the complete j-ary tree with ℓ levels (labelled so that level i has j^{i-1} vertices), rooted at a vertex v (Figure 3).

Lemma 4: For all $\delta \geq 0$ and integers $x \geq 1$, there exists a graph G and vertex v so that $\hat{k}_\delta(v) - k(v) = x - 1$.

Proof: First note that since $T_{j,\ell}$ is a tree, it is 1-degenerate. We show that the root vertex v has $\hat{k}_\delta$ estimators with unbounded error. For any vertex u with level number i in $[2, \ell - \delta - 1]$, every vertex not equal to v in u's δ-neighborhood has degree $j + 1$. As a result, $d(u) = \hat{k}_1(u) = \cdots = \hat{k}_{i-1} = j + 1$, and $\hat{k}_i = \cdots = \hat{k}_\delta = j$ (since v has degree j and will have propagated inwards). Thus, for any $\delta \leq \ell - 1$, we have $\hat{k}_\delta(v) - k(v) = j - 1$. ■

Lemma 5: For all $\delta \geq 0$ and integers $x \geq 1$, there exists a graph G and vertex v so that $\check{k}_\delta(v) - k(v) = x - 1$.

Proof: Consider the graph $T'_{j,\ell}$ created by adding vertices $w_1, \ldots, w_j$ to $T_{j,\ell}$ and then connecting each of them to each of the leaves of $T_{j,\ell}$ (see Figure 3). Then the root v is the vertex of minimum degree in $T'_{j,\ell}$ and has core number j. Any induced subgraph of $T'_{j,\ell}$ that does not include at least one w_i is a tree, making it 1-degenerate. Thus $k(v) - \breve{k}_\delta(v) = j - 1$ whenever $\ell \geq \delta + 1$. ■

Note that in $T_{j,\ell}$, $\breve{k}_\delta(v) = k(v)$ for any $\delta > 0$. Likewise, in $T'_{j,\ell}$, $\hat{k}_\delta(v) = k(v)$ for any $\delta \geq 0$. Despite the fact that the errors of both estimators can theoretically be arbitrarily large, structures causing egregious errors (like $T_{j,\ell}$ and $T'_{j,\ell}$) are unlikely to occur in real world networks; we provide evidence to support this claim in the next sections.

C. Expected behavior on random graphs

In order to better understand the errors generated when approximating core number with $\hat{k}_\delta$, we analyze its behavior on a well-studied random graph model.

Definition 7 ([13]): Erdös-Rényi random graphs, denoted $\mathcal{G}(n,p)$, are the family of graphs with n vertices constructed by placing an edge between each pair of vertices uniformly at random with probability p.

To avoid confusion, we use $\mathcal{G}(n,p)$ to denote the set of all Erdös-Rényi random graphs with n vertices and edge probability p and $G(n,p)$ for a specific instance. Since all graphs on n vertices occur in $\mathcal{G}(n,p)$ with non-zero probability (when $p \in (0,1)$), analysis typically focuses on whether a graph property is very likely (or unlikely) to occur as the size of an Erdös-Rényi random graph grows large. In keeping with prior work ([14]–[16]), we assume the average degree is fixed, letting $(n-1)p = \bar{d}$, a constant. Under this assumption, we using the following notion of "very likely":

Definition 8: A random event X is said to happen *asymptotically almost surely* (a.a.s.) if $\lim_{n\to\infty} \mathbb{P}[X] = 1$.

Specifically, we focus our attention on the growth of the error term $\hat{k}_1(v) - k(v)$ as $n \to \infty$ by deriving probabilistic expressions for $\hat{k}_1(v)$ and $k(v)$ for any $v \in G(n,p)$, then demonstrating how each term grows with n compared to a function in $\omega(1)$ (recall a function $f(n)$ is $\omega(1)$ if $\lim_{n\to\infty} \frac{1}{f(n)} = 0$).

Theorem 4: Suppose $\epsilon(n)$ is $\omega(1)$. Then for any $v \in G(n,p)$, $\hat{k}_1(v) - k(v)$ is $O(\epsilon(n))$ asymptotically almost surely.

Proof: Fix a vertex v, and let $S_{>\kappa}$, $S_{<\kappa}$, and $S_{=\kappa}$ be defined to be the subsets of $N_1(v)$ with vertices of degree greater than, equal to, or less than κ, respectively. We first evaluate $\mathbb{P}[\hat{k}_1(v) = \kappa | d(v) = d]$. By Definition 6, if $\hat{k}_1(v) = \kappa$, v has at least κ neighbors u with $\hat{k}_0(u) \geq \kappa$ but less than $\kappa+1$ with $\hat{k}_0(u) \geq \kappa+1$ (or else $\hat{k}_1(v) > \kappa$). Therefore, $\hat{k}_1(v) = \kappa$ implies $|S_{>\kappa}| \leq \kappa$ and $|S_{=\kappa}| + |S_{>\kappa}| \geq \kappa$, and

$$\mathbb{P}[\hat{k}_1(v) = \kappa | d(v) = d] = \sum_{i=0}^{\kappa} \sum_{j=0}^{d-\kappa} \frac{d!}{i!j!x!} \mathbb{P}[(|S_{>\kappa}| = i) \wedge (|S_{<\kappa}| = j) \wedge (|S_{=\kappa}| = x)],$$

where $x = d - i - j$.

As n tends to infinity, the probability that any two neighbors of v have an edge between them approaches 0. Therefore, the degrees of v's neighbors can be treated as independent, identical distributions in the limit. Since n is large and $\bar{d}$ is fixed, this distribution is asymptotically Poisson with mean $\bar{d}$. If $\zeta_\lambda(k)$ and $Z_\lambda(k)$ denote the Poisson probability mass function and cumulative distribution function, respectively, with mean λ (evaluated at k), then:

$$\mathbb{P}[\hat{k}_1(v) = \kappa | d(v) = d] = \sum_{i=0}^{\kappa} \sum_{j=0}^{d-\kappa} \frac{d!}{i!j!x!} (1 - Z_{\bar{d}}(\kappa-1))^i Z_{\bar{d}}(\kappa-2)^j \zeta_{\bar{d}}(\kappa-1)^x. \quad (1)$$

By computing Equation 1 at each value of d for which κ is a possible value for the core number, we have:

$$\mathbb{P}[\hat{k}_1(v) = \kappa] = \sum_{d=\kappa}^{n-1} \zeta_{\bar{d}}(d) \cdot \mathbb{P}[\hat{k}_1(v) = \kappa | d(v) = d]. \quad (2)$$

Pittel et al. [14] demonstrated that in $\mathcal{G}(n,p)$, the proportion of vertices in the k-core is a.a.s. a function of $\bar{d}$ but not of n. Moreover, for any vertex v, $k(v)$ is a.a.s. bounded by a constant (equivalently, in $\Theta(1)$). If $\hat{k}_1(v)$ were also bounded by a constant, then the error term $\hat{k}_1(v) - k(v)$ would be a.a.s. $O(1)$. However, since the Poisson random variables in Equations 1 and 2 are only parameterized by $\bar{d}$ and not by n, the proportion of vertices in $G(n,p)$ with $\hat{k}_1 = \kappa$ is a.a.s. convergent to some non-zero constant. Thus, the probability of having an arbitrarily large value of $\hat{k}_1$ does not vanish as n grows large for a fixed (constant) κ.

Let κ be a function of n in $\omega(1)$. By Stirling's approximation of the factorial,

$$\zeta_{\bar{d}}(\kappa) \approx \frac{e^{-\bar{d}}}{\sqrt{2\pi\kappa}} \left(\frac{e\bar{d}}{\kappa}\right)^{\kappa}.$$

Then as n grows large, $\zeta_{\bar{d}}(\kappa) \to 0$ and $Z_{\bar{d}}(\kappa) \to 1$. In Equation 1, the probability that a neighbor u of vertex v has degree at least κ (that is, $u \in S_{>\kappa} \cup S_{=\kappa}$) is asymptotically zero, and consequently $\mathbb{P}[\hat{k}_1(v) = \kappa]$ also vanishes in the limit. This implies that a.a.s. $\mathbb{P}[\hat{k}_1(v) = \kappa] \in O(\epsilon(n))$ for any error function $\epsilon(n) \in \omega(1)$. Using the result of [14] that $k(v) \in \Theta(1)$, we have a.a.s. $\hat{k}_1(v) - k(v) \in O(\epsilon(n))$. ■

IV. Experimental Results

In the previous section, the behavior of the propagating estimator $\hat{k}_\delta$ was analyzed from a theoretical perspective. In order to enhance this picture, we present computational results on a corpus of real data.

A. Methods

The estimators $\hat{k}_\delta$ and $\breve{k}_\delta$ were evaluated on nine real-world graphs that appear in the following section[3]. Not only do these graphs cover a variety of domains, but they also are structurally dissimilar. The graphs vary in size, density, core structure, and diameter (see Table I and Figure 4).

[3]We also tested several additional graphs, which gave qualitatively similar results, and are thus omitted for length. The results can be found in the arXiv version of this paper.

Graph	$\|V\|$	$\|E\|$	$\max d(v)$	D	Δ
AMAZON [17] Co-purchases	334863	925872	549	6	47
AS [17] Autonomous systems	6214	12232	1397	12	9
CA-ASTROPH [17] Academic citations	17903	196972	504	56	14
DBLP [17] Academic citations	317080	1049866	343	113	23
ENRON [17] Email correspondence	33696	180811	1383	43	13
FACEBOOK [18] Facebook friendship	36371	1590655	6312	81	7
GNUTELLA [17] Peer-to-peer filesharing	26498	65359	355	5	11
H. SAPIENS [19] Protein-protein interation	18625	146322	9777	47	10
WPG [20] Western US power grid	4941	6594	19	5	46

TABLE I: Summary statistics for real-world graphs

We computed the core number $k(v)$, as well as the values of $\hat{k}_\delta(v)$ and $\breve{k}_\delta(v)$ for each vertex[4], letting δ vary from 0 to Δ. To compare the accuracy of the estimators among vertices, we normalize by the true core number at each vertex. We refer to this metric as the *core number estimate ratio*. When the estimator ($\hat{k}_\delta$ or $\breve{k}_\delta$) is exactly equal to the core number, the core number estimate ratio is 1, its *optimal* value. Since $\hat{k}_\delta$ is an upper bound on k, its core number estimate ratio is always at least one and becomes *less optimal* the larger it gets; the opposite is true for $\breve{k}_\delta$, a lower bound.

B. Results

We first turn our attention to how often the estimators achieve optimal core number estimate ratios. Figure 5 shows how the proportion of vertices with ratio one grows as δ increases from zero to four. In all the graphs, the core number estimate ratio for $\hat{k}_\delta$ is optimal at least as often as that of $\breve{k}_\delta$ at $\delta \leq 1$. Additionally, the proportion of vertices with optimal $\hat{k}_\delta$ estimate ratios is large in all the graphs (often upwards of 90%). While the propagating estimator does not have as pronounced of an advantage over the induced estimator when $\delta = 2$, the number of vertices with optimal $\hat{k}_\delta$ estimate ratios still grows noticeably.

We also examined the distribution of core number estimate ratios among those vertices where the estimate was not exact. The change in this distribution over the range $\delta = 1$ to $\delta = 4$ is shown in Figure 6, demonstrating that not only are the sub-optimal estimates closely centered around 1, but also that increasing δ can significantly decrease the size of the "tail" of the distribution (thereby improving the core number estimates of those vertices with the least optimal ratios).

Although Figures 5 and 6 suggest that $\hat{k}_\delta$ and $\breve{k}_\delta$ can accurately estimate the core numbers in real world graphs using only knowledge of the δ-neighborhood with a small value of δ, it is important to understand how the size of the δ-neighborhood impacts the behavior of the estimates. The purpose of having a localized estimate is to reduce the size of the input needed to compute the core number of a vertex. If the average δ-neighborhood encompasses most of the graph, then not only is this purpose defeated, but we also may not be able

[4]Code and data are available at `https://dl.dropboxusercontent.com/u/32167511/core_number_estimate.zip`

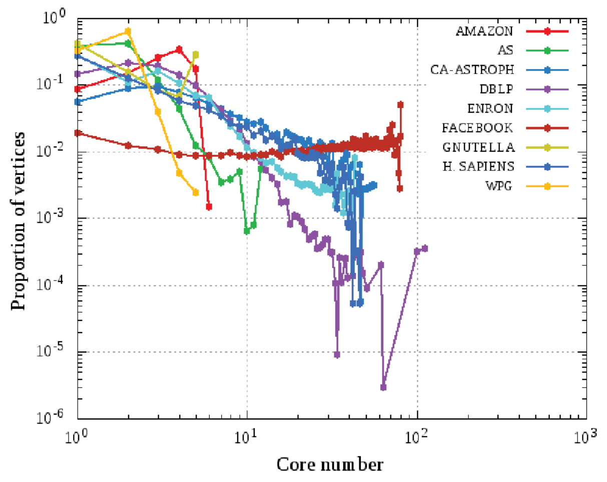
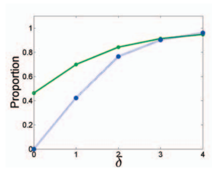


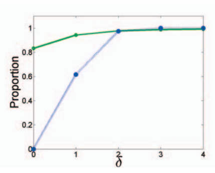
Fig. 4: Core number distribution for the real world networks in Table I.

to judge whether the accuracy of the localized estimates is only due to having knowledge of the entire graph (as opposed to any theoretical merits of the algorithms themselves). The mean and variance of the proportion of vertices in the δ-neighborhood is shown in Table II. The rate of growth of δ-neighborhood sizes varies significantly among the nine graphs, which suggests that picking a value of δ to maintain appropriately small δ-neighborhoods is highly dependent on the structure of the graph. Nonetheless, the average size of the δ neighborhood is below ten percent of the entire graph for all datasets at $\delta = 1$ and in all but one (namely FACEBOOK, which we know to be significantly different from the other networks) at $\delta = 2$.

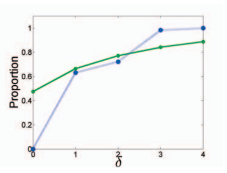
Another natural way to measure the relative amount of



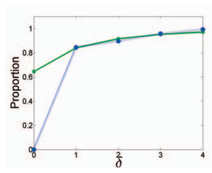
(a) AMAZON



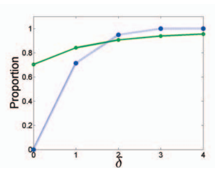
(b) AS



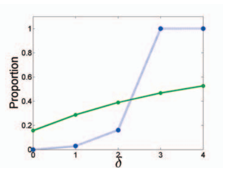
(c) CA-ASTROPH



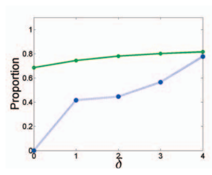
(d) DBLP



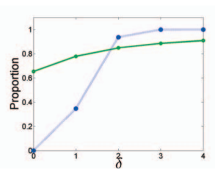
(e) ENRON



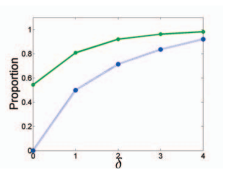
(f) FACEBOOK



(g) GNUTELLA



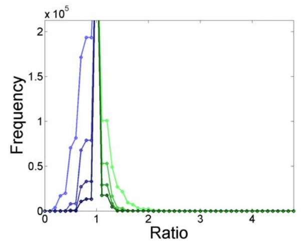
(h) H. SAPIENS

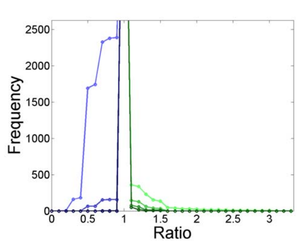


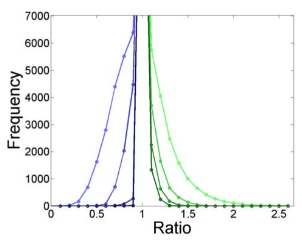
(i) WPG

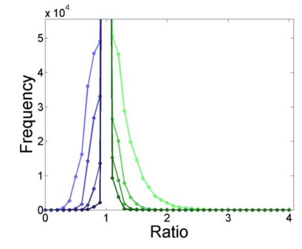
Fig. 5: Proportion of vertices with optimal core number estimate ratios for the propagating (solid green) and induced (dashed blue) estimators as a function of δ.

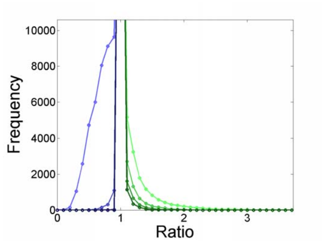
information in the δ-neighborhood is to normalize by the

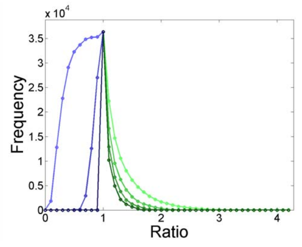

(a) AMAZON

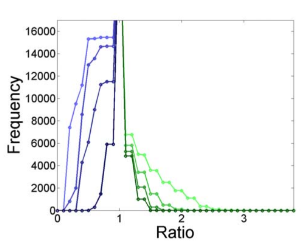

(b) AS

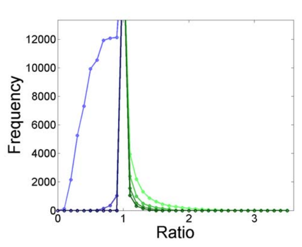

(c) CA-ASTROPH


(d) DBLP


(e) ENRON


(f) FACEBOOK


(g) GNUTELLA


(h) H. SAPIENS

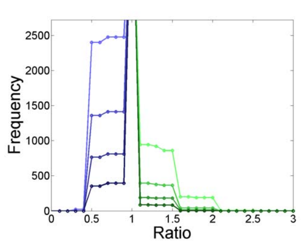

(i) WPG

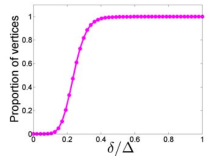
Fig. 6: Number of vertices with core number estimate ratios less optimal that a given threshold from $\delta = 1$ (lightest line) to $\delta = 4$ (darkest line). $\hat{k}_\delta$ is shown in green while $\breve{k}_\delta$ is shown in blue. Because the number of vertices with optimal ratios is frequently large (see Figure 5), the vertices with optimal ratios may not appear within the limits of the plot in order better capture the distribution of those vertices with suboptimal ratios.

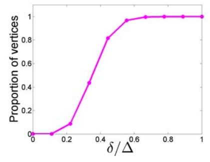
	$\delta = 1$		$\delta = 2$		$\delta = 3$		$\delta = 4$	
Graph	**Avg.**	**Var.**	**Avg.**	**Var.**	**Avg.**	**Var.**	**Avg.**	**Var.**
AMAZON	.00	.00	.00	.00	.00	.00	.00	.00
AS	.00	.00	.09	.01	.44	.07	.82	.04
CA-ASTROPH	.00	.00	.03	.00	.25	.04	.66	.06
DBLP	.00	.00	.00	.00	.00	.00	.03	.00
ENRON	.00	.00	.03	.00	.28	.04	.74	.06
FACEBOOK	0.00	0.00	0.21	0.03	0.89	0.02	1.00	0.00
GNUTELLA	.00	.00	.00	.00	.02	.00	.15	.02
H. SAPIENS	0.00	0.00	0.31	0.07	0.81	0.04	0.98	0.00
WPG	.00	.00	.00	.00	.00	.00	.01	.00

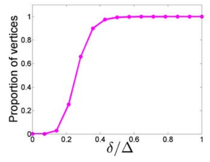
TABLE II: Proportion of vertices in N_δ. Values less than .01 rounded to 0.

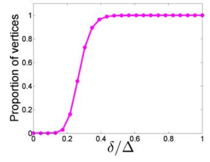
diameter Δ. Figure 7 shows that the average proportion of vertices in the δ-neighborhood is approximately uniform in all graphs when δ is expressed as a fraction of Δ. In particular, there is a significant increase in the rate of growth of the δ-neighborhood size occurring when δ is approximately 20% of the diameter. Thus, as one might expect, neighborhood-based core number estimates seem most appropriate when δ is a small fraction of the diameter.

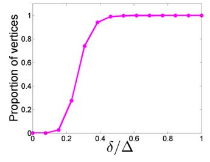
To see the effect of this normalized setting on accuracy, consider Figure 8. After normalizing δ by the diameter, the variation between graphs is less pronounced. Even when δ is small compared to Δ, the optimum core number estimate ratio can be achieved. It is worthy to note that FACEBOOK is a stark exception in which a significant proportion of vertices cannot acheive an optimal core number estimate ratio even when $\delta = \Delta$. This graph has many vertices with $d(v) \gg k(v)$ and a small diameter. Thus, many vertices have very inaccurate $\hat{k}_0$-values, which propagate inwards and remain uncorrected due to the small number of refinements performed on the estimate. Ultimately, we conclude that $\hat{k}_\delta$ best achieves its goal of accurately estimating the core number using only a small local section of the graph when the graph has a large diameter and the ratio δ/Δ is small (e.g. less than 0.2).

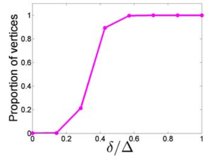

(a) AMAZON

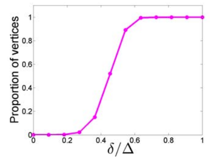

(b) AS

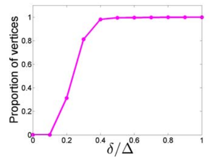

(c) CA-ASTROPH


(d) DBLP


(e) ENRON


(f) FACEBOOK


(g) GNUTELLA


(h) H. SAPIENS

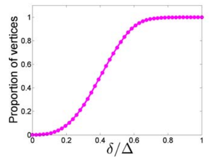

(i) WPG

Fig. 7: Average proportion of vertices in N_δ as a function of δ/Δ.

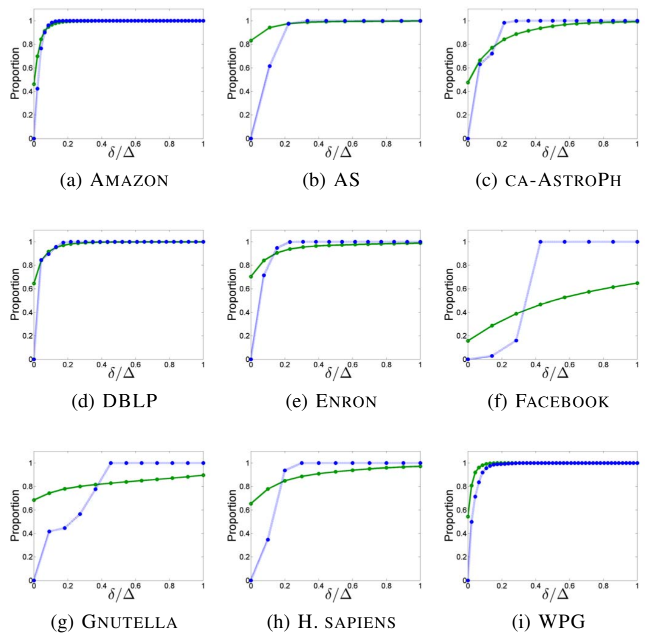


Fig. 8: Proportion of vertices with optimal core number estimate ratios for the propagating estimator (solid green) and the induced estimator (dashed blue) as a function of δ. The x-axis has been normalized by the diameter.

V. Application to Network Experimentation

We now turn to the domain of network experiments and use the $\hat{k}_\delta$ estimator to address an open problem given in [7].

A. Problem Statement

Recall from the introduction that a *network treatment experiment* is a random experiment in which some subjects are given a treatment and the rest are not. It differs from other experiments in that the effects of the treatment are assumed to be dependent on interactions between subjects, which can be modeled by a graph. The general goal is to measure the subjects' experiences in a hypothetical universe where the entire graph is treated by observing the experience of the subject when only some of the graph is treated. Ugander et al. [7] focused on local properties of the vertices to compare these two scenarios. In particular, they identified two useful ways to concretely measure the experience of a subject via graph properties:

Definition 9 ([7]): A vertex v experiences *absolute k-degree exposure* if v and at least k of v's neighbors receive treatment.

Definition 10 ([7]): A vertex v experiences *absolute k-core exposure* to a treatment condition if v belongs to the k-core of the graph $G[V']$, where V' is the set of treated vertices.

We will use $X_k^{(d)}(v)$ and $X_k^{(c)}(v)$ to denote the events that a vertex v experiences absolute k-degree and absolute k-core exposure, respectively.

In order to reduce variance in later sections of their analysis, Ugander et al. first cluster the graph and then assign treatment randomly to the clusters (as opposed to individual subjects). If a cluster is chosen to be treated, all vertices in the cluster receive treatment; otherwise none of them do. Ugander et al. utilize a 3-*net clustering* that is formed by growing balls of radius two centered at randomly selected vertices until every vertex is covered by some ball. The procedures for computing the probabilities of $X_k^{(d)}(v)$ and $X_k^{(c)}(v)$ are independent of the method by which the graph was clustered, so we choose to omit further detail here and refer the reader to [7] for details.

Once the graph is clustered, a recursive function can be used to compute the probability that vertex v experiences absolute k-degree exposure. We follow the notation of [7]. Let s be the number of clusters that contain at least one vertex in $\{v\} \cup N_1(v)$, indexed $\{1, \ldots, s\}$ so that v resides in the highest numbered cluster. If p is the probability that a cluster is treated and $\vec{w_v} = (w_{v,1}, \ldots w_{v,s})$ is the number of edges from v to the vertices in each cluster, then

$$\mathbb{P}[X_\kappa^{(d)}(v)] = pf(s-1, \kappa - w_{v,s}; p, \vec{w_v}), \tag{3}$$

where the function $f(j, T; p, \vec{w_v})$ is defined as

$$\begin{aligned} f(1, 0; p, \vec{w_v}) &= 1 \\ f(1, T; p, \vec{w_v}) &= p\mathbf{1}[T \leq w_{v,1}] \\ f(j, T; p, \vec{w_v}) &= pf(j-1, T - w_{v,j}; p, \vec{w_v}) \\ &\quad + (1-p)f(j-1, T; p, \vec{w_v}), \end{aligned}$$

where $\mathbf{1}[B]$ denotes the *indicator function* (evaluates to 1 if the Boolean expression B is true and 0 otherwise).

The function f defined above recursively visits each cluster j containing a neighbor of v and considers the probability that v is T-degree exposed in the first j clusters conditioned on whether cluster j receives treatment. If j is treated, v needs to have $T - w_{v,j}$ treated neighbors in the first $j-1$ clusters; otherwise, it needs T such neighbors. It follows from Definition 9 that if v is k-degree exposed, cluster s is necessarily treated. This also implies that all of v's neighbors in the same cluster are necessarily treated as well. Thus, we are ultimately concerned with finding $\kappa - w_{v,s}$ treated neighbors in the remaining $s-1$ clusters that contain a neighbor of v. Using dynamic programming, we can compute $\mathbb{P}[X_i^{(d)}(v)]$ for all $0 \leq i \leq \kappa$ in $O(s\kappa)$ time.

B. Estimating k-core exposure probabilities

In [7], Ugander et al. left computing the exact probability of absolute k-core exposure as an open problem, since the core decomposition requires knowledge of the entire graph. They instead defer to the fact that the absolute k-core exposure probability is bounded from above by the absolute k-degree exposure probability and use the latter in lieu of the former. This is problematic because there may not be a consistent relationship between $d(v)$ and $k(v)$. For example, if v is a vertex with degree 100 and core number 10, $\mathbb{P}[X_{20}^{(c)}] = 0$ independent of $\mathbb{P}[X_{20}^{(d)}]$. Although this is an extreme case, there are more general cases where the two probabilities are not correlated. Specifically, vertices that require a large value of δ before $\check{k}_\delta(v) = k(v)$ (as in Figure 9) can have many treated neighbors without having a large core number. Recall that $\hat{k}_0(v)$ is the degree of v. As we have shown, even expanding the scope and computing $\hat{k}_1(v)$ can yield a considerably more accurate estimate of of the core number than the degree.

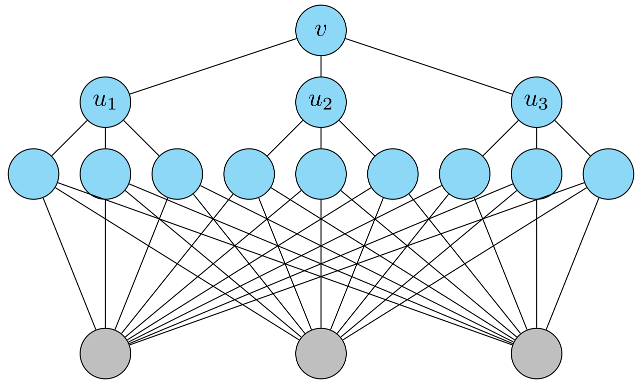


Fig. 9: $T'_{3,3}$ with 13 of 16 vertices treated. $k(v) = k(u_1) = k(u_2) = k(u_3) = 3$. Although v, u_1, u_2, and u_3 have all of their neighbors treated, they only have core number 1 with respect to the treated subgraph.

Therefore, a tighter bound of the core exposure probability can be achieved by examining the degree exposure probability of v's neighbors. To capture this, we introduce a $\hat{k}_1$-related condition we call *neighbor-degree exposure*.

Definition 11: A vertex v experiences *absolute k-neighbor-degree exposure* if at least k of v's neighbors experience absolute k-degree exposure.

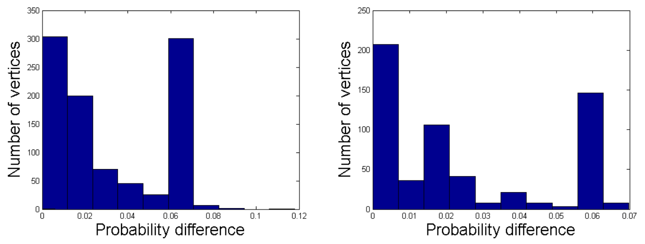
We denote the event that vertex v is absolute k-neighbor-degree exposed with $\hat{X}_k(v)$. Algorithm 3 gives a method for computing $\mathbb{P}[\hat{X}_\kappa(v)]$, which can then be used to estimate (specifically, find an upper bound on) $\mathbb{P}[X_\kappa^{(c)}(v)]$.

Algorithm 3 Absolute k-neighbor-degree exposure probability of v

```
    INPUT: Graph $G$, vertex $v$, clustering $C$, exposure probability $p$, desired exposure level $\kappa$
    OUTPUT: $\mathbb{P}[\hat{X}_\kappa(v)]$
 1: Let $C(x)$ denote the cluster containing $x$
 2: $\mathcal{C} \leftarrow \left(\bigcup_{u \in N_2(v)} C(u)\right) \setminus C(v)$
 3: $\hat{p}_\kappa \leftarrow 0$
 4: for $\mathcal{S} \subseteq \mathcal{C}$ do
 5:     $Y \leftarrow \{u \in N_1(v) : X_\kappa^{(d)}(v)$ is true in $G[\mathcal{S} \cup C(v)]\}$
 6:     if $|Y| \geq k$ then
 7:         $\hat{p}_\kappa \leftarrow \hat{p}_\kappa + p^{|\mathcal{S}|+1}(1-p)^{|\mathcal{C}|-|\mathcal{S}|}$
 8:     end if
 9: end for
10: return $\hat{p}_\kappa$
```

The algorithm iterates through all subsets of clusters containing a vertex in v's 2-neighborhood and determines whether treating them yields a scenario where v has k neighbors that are absolute k-degree exposed. If so, line 7 adds the probability of that configuration occurring to the final probability. Because it enumerates all possible subsets of $\mathcal{C}$, Algorithm 3 will run in $O(s\kappa \cdot d(v) \cdot 2^s)$ time in the worst case. While this algorithm is exponential in the number of clusters, Ugander et al. assume that the graph satisfies some restricted growth conditions[5]. In this case, the number of clusters that contain vertices from $N_2(v)$ does not grow with respect to the size of the graph [7] which bounds the running time at $O(\kappa \cdot d(v))$.

[5]Namely, $\exists c$ such that $|N_{\delta+1}(v)| \leq c \cdot |N_\delta(v)|\ \forall v \in V$



(a) $\kappa = 4$ (b) $\kappa = 5$

Fig. 10: $\mathbb{P}[X_\kappa^{(d)}] - \mathbb{P}[\hat{X}_\kappa]$ for the WPG graph at $p = 0.25$. Vertices with $\mathbb{P}[X_\kappa^{(d)}(v)] = 0$ are omitted.

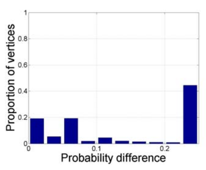
In graphs failing the restricted growth requirements, the running time can still be improved. Note that if treating a specific subset of clusters $\mathcal{S}$ on line 4 does not yield κ vertices in $N_1(v)$ that are κ-degree exposed, then treating any $\mathcal{S}' \subseteq \mathcal{S}$ also cannot yield at least κ vertices in $N_1(v)$ that are κ-degree exposed. Thus, if the subsets of $\mathcal{C}$ are enumerated in decreasing order of their sizes, we can prune the search space to avoid needless computation. Moreover, the clustering algorithm can be biased towards selecting 3-net clusterings that minimize $|\mathcal{C}|$. For example, one possible bias would be to select the centers of the balls with probability proportional to their degrees.

We applied Algorithm 3 and Equation 3 to the WPG data set and binned the data based on the difference $\mathbb{P}[\hat{X}_\kappa] - \mathbb{P}[X_\kappa^{(d)}]$ as shown in Figure 10. It is particularly noteworthy that multiple vertices have a neighbor-degree exposure probability of zero but a non-zero probability of degree exposure. Moreover, many of those vertices have their degree exposure probability maximized (equal to 0.25). Thus, the empirical data confirms that absolute degree exposure probability may be a misleading estimate of absolute core exposure probability.

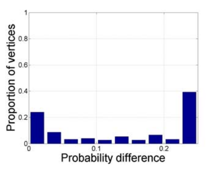
Finally, we consider a second approach for improving the approximation of $\mathbb{P}[X_k^{(c)}(v)]$ that, like $\hat{k}_\delta$, tightens an upper bound on $\mathbb{P}[X_\kappa^{(c)}(v)]$ by using the bounds on $\mathbb{P}[X_\kappa^{(c)}(u)]$ for u in $N_1(v)$. We examine those vertices u' in $N_1(v)$ that satisfy $\mathbb{P}[X_\kappa^{(d)}(u')] = 0$. These vertices cannot contribute to $\mathbb{P}[\hat{X}_\kappa(x)]$, so we can disregard them when computing $\vec{w_v}$. Thus, we can use Equation 3 with a modified $\vec{w_v}$ to get a tighter upper bound on $\mathbb{P}[X_\kappa^{(c)}]$. Figure 11 shows that pruning can decrease the probability of a majority of the vertices (in fact, many probabilities decrease from p to 0). This further bolsters our argument that $X_\kappa^{(c)}(v)$ is only weakly correlated with $X_\kappa^{(d)}(v)$, but using information from v's neighbors can yield a much tighter upper bound at minimal additional cost.

VI. Conclusions and Future Work

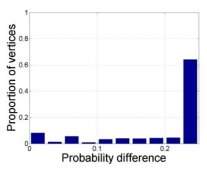
We introduced $\hat{k}_\delta$, a novel method of estimating the core number of a vertex that uses only the data available in the δ-neighborhood of the vertex. We formally proved that in an Erdös-Rényi graph, the error of $\hat{k}_1$ grows arbitrarily slowly with respect to the size of the graph. After computing $\hat{k}_2$ on a representative corpus of real-world networks, we demonstrated that a high-accuracy estimate of the core number can be achieved using a limited subset of the graph. Finally, we



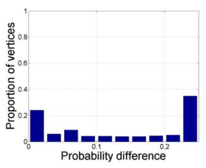
(a) AMAZON



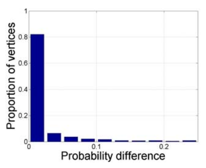
(b) AS



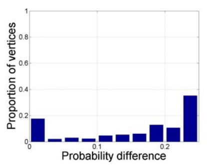
(c) CA-ASTROPH



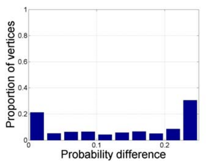
(d) DBLP



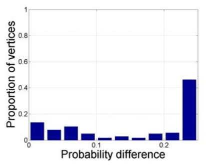
(e) ENRON



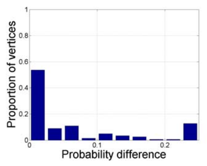
(f) FACEBOOK



(g) GNUTELLA



(h) H. SAPIENS



(i) WPG

Fig. 11: Histogram of differences between $\mathbb{P}[X_\kappa^{(d)}]$ before and after pruning for $\kappa = 7$ and $p = 0.25$. The x-axis gives the difference in probability, while the y-axis gives the proportion of vertices occurring in that bin. For clarity, only those vertices which are not pruned are considered in the plot.

described two ways in which the estimators could be used to improve calculations in network treatment experiments.

There are a number of natural extensions to this research. Algorithm 2 computes $\hat{k}_{\delta-1}$ for each neighbor u_i of v, which in turn requires calculating $\hat{k}_{\delta-1}$ and so forth. However, since $\hat{k}_\delta(v)$ is geometrically the value at the intersection of the functions $d(v) - i + 1$ and $k_{\delta-1}(u_i)$, refining the core number estimates at the "first" vertices $(u_1, u_2, \dots)$ and "last" vertices $(u_d, u_{d-1}, \dots)$ may not affect where the curves intersect. Thus, computational complexity could possibly be reduced by only refining the estimates of vertices near u_i.

There may also be use for $\hat{k}_\delta$ in graph property testing. Property testing refers to using an easily computable graph property in order to give an estimate of a less tractable property. For example, the *hyperbolicity* of a graph informally measures the extent to which a graph is tree-like [21]. As was discussed in Section III-B, tree-like structures with high degree but low degeneracy can lead to large errors in $\hat{k}_\delta$. Therefore, a large error in $\hat{k}_\delta(v)$ may indicate that v participates in a structure with low hyperbolicity. Since the hyperbolicity is computed in $O(|V|^4)$ time, it would be significantly faster to indirectly flag such vertices by computing $\hat{k}_\delta(v)$ and $k(v)$ at every vertex and observing their difference.

ACKNOWLEDGMENTS

The authors thank Johan Ugander for introducing them to the problem of calculating the k-cores in a network experiment during a workshop at the Statistical and Applied Mathematical Sciences Institute (SAMSI) and for providing helpful comments and discussion that improved the manuscript. This work was supported in part by the National Consortium for Data Science Faculty Fellows Program and the Defense Advanced Research Projects Agency under SPAWAR Systems Center, Pacific Grant N66001-14-1-4063. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of DARPA, SSC Pacific, or the NCDS.

REFERENCES

[1] V. Batagelj and M. Zaversnik, "An O(m) algorithm for cores decomposition of networks," *CoRR*, 2003.

[2] C. Giatsidis, D. M. Thilikos, and M. Vazirgiannis, "Evaluating cooperation in communities with the k-core structure," in *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*. IEEE, 2011, pp. 87–93.

[3] D. W. Matula and L. L. Beck, "Smallest-last ordering and clustering and graph coloring algorithms," *J.ACM*, vol. 30, no. 3, pp. 417–427, jul 1983.

[4] M. Kitsak, L. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. Stanley, and H. Makse, "Identification of influential spreaders in complex networks," *Nature Physics*, vol. 6, no. 11, pp. 888–893, Aug 2010.

[5] G. Bader and C. W. V. Hogue, "An automated method for finding molecular complexes in large protein interaction networks," *BMC Bioinformatics*, vol. 4, no. 1, pp. 1–27, 2003.

[6] J. I. Alvarez-Hamelin, A. Barrat, and A. Vespignani, "Large scale networks fingerprinting and visualization using the k-core decomposition," in *Advances in Neural Information Processing Systems 18*. MIT Press, 2006, pp. 41–50.

[7] J. Ugander, B. Karrer, L. Backstrom, and J. M. Kleinberg, "Graph cluster randomization: network exposure to multiple universes," *CoRR*, 2013.

[8] A. E. Saríyüce, B. G., G. Jacques-Silva, K. Wu, and U. V. Çatalyürek, "Streaming algorithms for k-core decomposition," *Proc.VLDB Endow.*, vol. 6, no. 6, pp. 433–444, apr 2013.

[9] A. Montresor, F. D. Pellegrini, and D. Miorandi, "Distributed k-core decomposition," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 2, pp. 288–300, 2013.

[10] P. Jakma, M. Orczyk, C. S. Perkins, and M. Fayed, "Distributed k-core decomposition of dynamic graphs," in *Proceedings of the 2012 ACM conference on CoNEXT student workshop*, ser. CoNEXT Student '12. New York, NY, USA: ACM, 2012, pp. 39–40.

[11] R. H. Li and J. X. Yu, "Efficient core maintenance in large dynamic graphs," *CoRR*, 2012.

[12] J. Cheng, Y. Ke, S. Chu, and M. T. Ozsu, "Efficient core decomposition in massive networks," in *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, ser. ICDE '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 51–62.

[13] P. Erdös and A. Rényi, "On the evolution of random graphs," *Publ.Math.Inst.Hung.Acad.Sci*, vol. 5, pp. 17–61, 1960.

[14] B. Pittel, J. Spencer, and N. Wormald, "Sudden emergence of a giant k-core in a random graph," *Journal of Combinatorial Theory, Series B*, vol. 67, no. 1, pp. 111–151, 5 1996.

[15] S. Janson and M. J. Luczak, "Asymptotic normality of the k-core in random graphs," *The Annals of Applied Probability*, vol. 18, no. 3, pp. 1085–1137, 06 2008.

[16] T. Luczak, "Size and connectivity of the k-core of a random graph," *Discrete Math.*, vol. 91, no. 1, pp. 61–68, jul 1991.

[17] "Stanford large network dataset collection." [Online]. Available: http://snap.stanford.edu/data/

[18] A. L. Traud, P. J. Mucha, and M. A. Porter, "Social structure of Facebook networks," *Physica A*, vol. 391, pp. 4165–4180, 2012.

[19] "Biological general repository for interaction datasets." [Online]. Available: http://www.thebiogrid.org

[20] "Ilab interdisciplinary research institute." [Online]. Available: http://www.ilabsite.org/?page_id=12

[21] N. Cohen, D. Coudert, and A. Lancin, "Exact and approximate algorithms for computing the hyperbolicity of large-scale graphs," Laboratoire de Recherche en Informatique, Tech. Rep., 2012-09-25 2012, iD: hal-00735481, version 4.