Robust Adjacent Attributed Community Search

Niu Yudong
ydniu.2018@phdis.smu.edu.sg
Singapore Management University

1 INTRODUCTION

Attributed graphs(networks) have emerged as a prevalent model for representing relationships between entities with certain properties. For example, the coauthor relationships between scientists can be represented through a collaboration network and different scientists have different areas of expertise, which can be represented as attributes of nodes. Figure 1 shows an example of a collaboration network for computer scientists.

Identifying communities, which are groups of vertices that are densely connected and have homogeneous attributes, is crucial for understanding the structure of attributed networks as well as for applications such as recommendation. Thus, attributed community detection has been extensively studied in the literature. More recently, a related but different problem called attributed community search has been proposed. It is motivated by the need to make answers more meaningful and personalized to a specific user. For the given query node and attributes, attributed community search seeks to find the community that ***contains the query node*** as well as has ***high consistency with the query attributes***. For instance, consider a query proposed by user q_0 with query attribute {DB} in Figure 1, according to the definition of *attributed community search*, subgraph H_0 will be returned.

However, we have observed that under certain circumstances, the definition of *attributed community search* can lead to unsatisfactory results. To explain, suppose that user q_0, who mainly works on database, is now interested in database tuning through machine learning methods. Therefore, q_0 proposed a query with attribute {ML} to find a group of experts on machine learning to collaborate with. Intuitively, subgraph H_1 is a feasible result for this query. The reason is two folded. Firstly, H_1 is well-connected and attribute homogeneous, thus is indeed an attributed community. Secondly, H_1 is quite close to q_0, which means that the user can get in touch with this community easily. Nevertheless, *attributed community search* returns subgraph $H_0 \cup H_1$ to q_0, which has low attribute homogeneity. The root cause of this unsatisfactory result is that *attributed community search* pre-assumes that the query user is within a community that is homogeneous with the query attributes and thus requires the query node to be contained in the result subgraph in its definition.

We refer to queries that are analogous to the one in the above example as *exterior queries*. The query node of an exterior query is not contained in any community that is homogeneous with the query attributes. Analogously, we call queries that have the query nodes within a community that is homogeneous with the query attributes as *interior queries*. In real-world applications, it can be expected that exterior queries are more prevalent than interior queries. Still take the researcher collaboration network as an example. On the one hand, interior queries often do not provide much useful information to the query user. The reason is that given the user is within the target community, he should have already been familiar with other researchers within this research domain and thus doesn't rely on *attributed community search* to get the community anymore. On the other hand, with the fast development of various crossing subjects during recent years, it's becoming more and more essential for researchers to have good communication and collaboration with researchers outside his domain. At this point, the user will need the assistance from community search methods to find attributed communities of the research domain that he is interested in. Consequently, it's indispensable to adapt the definition of *attributed community search* so that exterior queries can be handled properly.

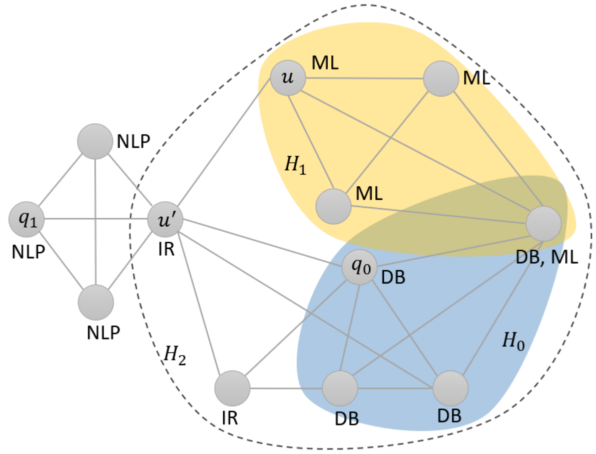

Figure 1: An example of a collaboration network for computer scientists.

To this end, we propose the problem *adjacent attributed community search* in this paper. Instead of forcing the result subgraph to contain the query node, *adjacent attributed community search* relaxes this constraint and only requires that the result subgraph is not far from the query node.

One may wonder whether existing *attributed community search* methods [15][13] can be adapted to solve *adjacent attributed community search*. The natural idea is to first identify a node v_{in} that is close to the query node and within a community w.r.t. query attributes and then run *attributed community search* methods from v_{in} to find the community. However, the difficulty lies in identifying a good v_{in}. In order to evaluate whether the community found from a given node is satisfactory, you actually have to first obtain the community, i.e. run *attributed community search* methods from the given node. Thus, in the worst case, N_{QA} times of *attributed community search* are required before finding a satisfactory community, where N_{QA} is the number of nodes having query attributes. Obviously, the time complexity is unacceptable. On the other hand, if we only obtain communities from a limited number k of nodes for the sake of efficiency, the method will become non-robust to possible errors in the network data. For example, suppose that $k = 1$ and q_1 proposes a query with attribute {ML} in Figure 1. The method will first find node u, which is the closest node to q_1 among nodes with attribute "ML", and then obtain the subgraph H_1 by *attributed community search*, which is satisfactory. However, if there is an error in the attributes data of the network, which causes the attribute of u' turns to "ML". Then, instead of finding u, the method will find u' and then obtain subgraph H_2, which is unsatisfactory obviously. This examples shows that the method that only examine limited number of nodes are non-robust even to a small error in the network data. Thus, a simple extension of *attributed community search* methods cannot achieve efficiency and robustness simultaneously.

This raises the following two major challenges for *adjacent attributed community search*. Firstly, given that there can be tension between the cohesiveness of the community with the closeness of the community to the query node, how should we combine these two goals while evaluating the result? Secondly, how can the query processing algorithm achieve efficiency and robustness at the same time?

While tackling these challenges, we make the following contributions in this paper:

- We formally define the problem of *adjacent attributed community search* by giving a score function which achieves a good combination of the cohesiveness and the closeness to query node of the candidates communities.
- We analyze the definition of *adjacent attributed community search* and show that it is non-monotone, non-submodular and non-supermodular, which signal huge computational challenges. We also formally prove that the problem is NP-hard.
- We develop a algorithm to solve *adjacent attributed community search* problem.
- We conduct extensive experiments on real world datasets, which show that our algorithm can efficiently and effectively find ground-truth communities and is robust to errors in network data.

We discuss related work in Section 2, and conclude the paper with a summary in Section 6.

2 RELATED WORK

Community Detection in Attributed Graphs. Community detection in attributed graphs aims at finding all densely connected components with homogeneous attributes. Existing methods mainly fall into three categories.

1) Topological-based clustering: The core idea of methods within this categories, including [21] [27] [26] [7], is to treat attributes of nodes as additional topological information, change the initial topology of the input graph accordingly and then use traditional clustering methods on the modified graph, which is now non-attributed.

2) Distance-based clustering: Methods belong to this category [6] [10] [11] computes a distance matrix between all pairs of nodes combining the topological and attributed information and then utilize classical distance-based clustering methods such as k-means to find communities.

3) Hybrid clustering: Hybrid clustering methods such as [17][24][25] first cluster the graph based on attributes and topological information separately and then merge the results properly.

A comprehensive description of these methods can be found in [12]. It's practically hard and inefficient to adapt these methods for *adjacent attributed community search* since they are inherently global and most of the work involved will be irrelevant to the community being searched.

Community Search. Community search on a graph aims at finding densely connected sub-graphs containing the query node. Various models based on different dense sub-graphs have been proposed: quasi-clique[8], densest subgraph[22], k-core[9][3][19] and k-truss[16]. These works focus on the structure of the community while ignoring node attributes, which will lead to results with poor homogeneity w.r.t. query attributes.

Recently, Yixiang Fang et al. and Xin Huang et al. proposed models for community search over attribute graphs based on attributed k-core[13] and (k,d)-truss with maximum attribute score [15][], respectively. However, these two methods cannot handle *exterior queries* and thus cannot be directly applied to *adjacent attributed community search*. In addition, intuitive extension of these two methods cannot achieve both efficiency and robustness as stated in the Introduction part.

Local Graph Partitioning. Instead of forcing the communities to comply with certain structural requirements, local graph partitioning aims at finding a low-conductance sub-graph containing the query node. These methods follow the framework of three steps:

(1) Compute a score for each node w.r.t. the query node.

(2) Sort all nodes in the graph according to the score and denote the subgraph induced by the first j nodes as S_j.

(3) Return S_j with smallest conductance as result.

Different scores have been studied:[20] proposed to use the probability for random walks from the query node reaching a node at the k-th step as the score, where k ranges from 1 to a large enough number. Thus, multiple rounds of the above procedure will be executed, which is not time efficient. Fung Chung et al. found that a single round of the procedure on Personalized Page-Rank value[1] or Heat Kernel value[5] w.r.t the query node is enough. [2][14] further achieves better (but probabilistic) running time through using the theory of evolving set.

Local graph partitioning only focuses on the conductance of the community and hasn't taken node attributes into consideration, thus cannot be applied over attributed graphs. At the meantime, these methods require that the query node itself is contained in a low conductance subgraph, which means they cannot deal with *exterior queries* even on non-attributed graphs. As a result, these methods cannot be applied to our problem directly.

3 PROBLEM DEFINITION

We consider an undirected, unweighted simple graph $G = (V, E)$ with $n = |V|$ nodes and $m = |E|$ edges. We use D and A to denote the *degree matrix* and *adjacent matrix* of G respectively. Each node v in G is attached with a set of attributes $\text{attr}(v)$. We use $V_\omega \subseteq V$ to denote the set of nodes having attribute ω, i.e., $V_\omega = \{v \in V | \omega \in \text{attr}(v)\}$. For a set of attributes Ω, let $V_\Omega = \bigcup_{\omega \in \Omega_q} V_\omega$. We say that node v is *attributed matched* w.r.t. Ω iff $v \in V_\Omega$.

We first give two definitions that will be used in our problem statement.

Definition 3.1. Given $c \in (0, 1)$, the *Personalized PageRank Vector* π_s w.r.t. node s is the vector satisfying the following equation:

$$\pi_s = c\chi_s + (1-c)\pi_s W$$

where χ_s is the indicator vector, i.e., $\chi_s(v) = 1$ if $v = s$ and 0 otherwise; $W = D^{-1}A$ is the random walk transition matrix of G.

Intuitively, $\pi_s(t)$ is the probability that a random walk from s terminates at t, if the random walk will terminate with probability c at each step. Vector π_s reflects the closeness of nodes in G w.r.t. node s.

Definition 3.2. The *conductance* of a subgraph $H = (V(H), E(H)) \subseteq G$ is

$$\phi(H) = \frac{|\partial(H)|}{\min(\mathrm{vol}(H), 2m - \mathrm{vol}(H))}$$

where $\partial(H) = \{(x, y) \in E | x \in V(H), y \notin V(H)\}$ and $\mathrm{vol}(H) = \sum_{x \in V(H)} \deg(x)$.

The value of $\phi(H)$ is a commonly accepted measurement for the structural cohesiveness of H.

Denoting a community query with query node v_q and query attributes set Ω_q as a binary tuple (v_q, Ω_q), we formally define the *adjacent attributed community search* problem.

Definition 3.3. *Adjacent attributed community search* aims at responding the query (v_q, Ω_q) with a subgraph H^* that satisfies the following conditions:
(1) Each node in H^* is *attributed matched* w.r.t. Ω_q.
(2) The value $\mathcal{S}(H^*) = |\ln \phi(H^*)| \cdot \min_{x \in V(H^*)} \pi(v_q, x)$ is maximized by H^* among subgraphs satisfying condition (1).

In this definition, condition (1) guarantees that H^* is attributively cohesive, and condition (2) ensures that H^* is structurally cohesive (note that $\mathcal{S}(H^*)$ is monotonically increasing with the decreasing of $\phi(H^*)$) as well as close to node v_q (the farthest node in $V(H^*)$ is required to be close to v_q). Also note that this definition does not require v_q to be contained in H^* and thus can handle *exterior queries* naturally.
We demonstrate the rationality of the definition of $\mathcal{S}(H^*)$ through proving some desirable properties of H^* caused by maximizing $\mathcal{S}(H^*)$.

Proposition 3.1. H^* is connected.

Proof. We prove this proposition through contradiction. Suppose that H^* is not connected, it can thus be partitioned into two parts H_1 and H_2 that is not connected. Note that both H_1 and H_2 satisfy condition (1) in the definition.
Without loss of generality, we assume that $\phi(H_1) \le \phi(H_2)$, then $\phi(H^*) \ge \phi(H_1)$. Also, $\min_{x \in H^*} \pi(v_q, x) \le \min_{x \in H_1} \pi(v_q, x)$ since $H_1 \subseteq H^*$. Thus,

$$\begin{aligned} \mathcal{S}(H^*) &= |\ln \phi(H^*)| \cdot \min_{x \in H^*} \pi(v_q, x) \\ &\le |\ln \phi(H_1)| \cdot \min_{x \in H_1} \pi(v_q, x) = \mathcal{S}(H_1) \end{aligned} \tag{1}$$

This is contradicted with the definition of H^*, which states that $\mathcal{S}(H^*)$ is maximized. Thus, H^* is connected. □

To describe the second property of H^*, we first define the *local non-matching coefficient* $\rho(v, \Omega)$ of node v w.r.t. the attributes set Ω. We denote the number of attributed matched neighbors w.r.t Ω of v as $\deg^+(v, \Omega)$ and the number of neighbors of v that are not attributed matched w.r.t. Ω as $\deg^-(v, \Omega)$. We define

$$\rho(v, \Omega) = \frac{\deg^-(v, \Omega) - \deg^+(v, \Omega)}{\deg(v)}$$

Proposition 3.2. Given a node v that is attributed matched w.r.t. Ω_q. If $\rho(v, \Omega_q) \ge \phi(H^*)$, then $v \notin H^*$.

Proof. We prove this proposition through contradiction. Suppose that $\rho(v, \Omega_q) \ge \phi(H^*)$ and $v \in H^*$. We use H^- to denote the subgraph obtained through removing v from H^*. Then, we have

$$\begin{aligned} \phi(H^-) &= \frac{|\partial H^*| + \deg_{in}(v) - \deg_{out}(v)}{\mathrm{vol}(H^*) - \deg(v)} \\ &\le \frac{|\partial H^*| + \deg^+(v, \Omega_q) - \deg^-(v, \Omega_q)}{\mathrm{vol}(H^*) - \deg(v)} \end{aligned} \tag{2}$$

where $\deg_{in}(v)$ and $\deg_{out}(v)$ represent the number of neighbors of v that are within and outside H^* respectively. We denote the right-side of inequality 2 as $\widehat{\phi}(H^-)$.
It’s not hard to see that the assumption $\rho(v, \overline{\Omega}_q) \ge \phi(H^*)$ is equivalent to $\widehat{\phi}(H^-) \le \phi(H^*)$. Combined with inequality 2, we get $\phi(H^-) \le \phi(H^*)$. We also have $\min_{x \in H^-} \pi(v_q, x) \ge \min_{x \in H^*} \pi(v_q, x)$ since $V(H^-) \subset V(H^*)$.
Thus, we have $\mathcal{S}(H^-) \ge \mathcal{S}(H^*)$. This is contradicted with the definition of H^*, which states that $\mathcal{S}(H^*)$ is maximized. Thus, $v \notin H^*$. □

Proposition 3.3. Suppose H° is a subgraph that only contains an attributed matched node v. Then, $\mathcal{S}(H^\circ) = 0$.

Proof. This proof is trivial since $\mathcal{S}(H^\circ) = |\ln \phi(H^\circ)| \cdot \pi(v_q, v) = |\ln 1| \cdot \pi(v_q, v) = 0$. □

4 C-SWEEP MODEL

5 EXPERIMENTS

5.1 Experimental Setup

Datasets. We consider several real world datasets including DBLP. In DBLP, a vertex denotes an author and an edge is a co-authorship relationship between two authors. For each author, we consider a binary node attribute that denotes whether he/she works on database related topics. The attribute is set to true if the author has papers published in SIGMOD, ICDE or VLDB, which are the three top conferences in the domain of database.
Ground-truth Communities. Since attribute ground-truth communities are not given in datasets such as DBLP, we use the communities that are derived from structural graph clustering [23][18][4] on attribute induced sub-graphs as ground-truth communities in our experiments. There are two crucial thresholds, μ and ε, that control the properties of communities derived from structural graph clustering. In our experiments, we set $\mu = 4$ and $\varepsilon = 0.5$.
Baselines. We compare our model with the state-of-the-art local attribute community search methods, K-core [13], Truss [15] and the local graph partitioning method Sweep [1].

K-core aims at finding maximal attribute matched k-core that containing the query node and Truss aims at finding (k, d)-truss that has highest attribute score and contains the query node. Both of the two methods require that the query node itself is equipped with the corresponding attribute and regard queries from unequipped nodes as bad queries. However, under our problem setting, it’s not necessary for the query node to have the corresponding attribute since the query node may not be contained in an attribute matched community. This obstacle makes the two methods not directly applicable to our problem. To avoid this obstacle, we first compute the personalized page-rank(PPR) value from the query node to all other nodes and select the attribute matched node with largest PPR value as the seed for K-core and Truss.

Sweep aims at finding local graph partition with low conductance. This method first compute the PPR value from the query node to all other nodes and use the sweep technique to produce a low conductance partition of the graph. However, this method doesn’t take attribute into consideration. To fix this problem, we modify Sweep so that only attribute matched nodes are considered during sweeping.

5.2 Results on Effectiveness

We first describe how we choose proper query nodes in our experiments. According to our problem setting, we should choose nodes that are close to but not within ground-truth attribute communities. Thus, we select nodes that are i-hop $i \in \{1, 2, 3\}$ from ground-truth attribute communities as our query nodes. Note that since the number of nodes that are i-hop from ground-truth communities may be very large, we randomly select 100 nodes for each i as our test samples.

Then, we have to select appropriate parameters for each method to continue the following experiments. All our methods contains the process of computing PPR values from query nodes, which depends on the parameter *teleportation*. We have observed that the value of *teleportation* only have slightly influence on the results of our three baselines. Thus, we will set the teleportation to 0.1 for all our baselines. On the other hand, the teleportation will have an significant influence on the effectiveness of our model Csweep, thus we will discuss the selection of teleportation for Csweep in the following. What’s more, we also have to choose suitable k and d for Truss.

Figure 3 illustrates the F1 score, precision and recall for Truss when the query nodes are 1-hop from the ground-truth communities for different k and d settings. Note that Truss may return *null* to queries if no (k, d)-truss can be found. Figure 2 shows the null-rate for Truss varying k. The null-rate of Truss increases along k and when k is larger than 7, the null-rate will be larger than 50%. Thus, we only show the results for k from 3 to 7 in Figure 3. It can be observed that Truss achieves the best result with $k = 3$ and $d = 2$. Thus, we set $k = 3$ and $d = 2$ as our default parameters for Truss in the following experiments. Figure 4 and Figure 5 illustrates the results for Truss when the query nodes are 2-hop and 3-hop from the ground-truth communities respectively. Similar observations about parameters k and d can be made from these two figures.

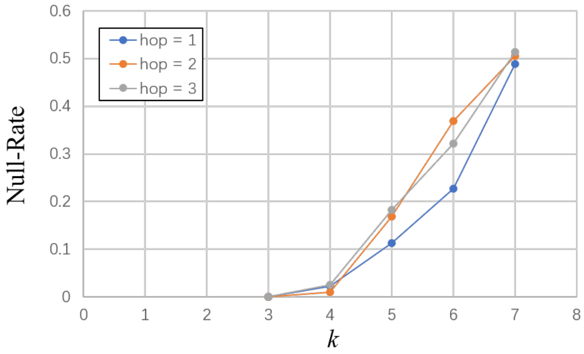

Figure 2: The null-rate of Truss varying k.

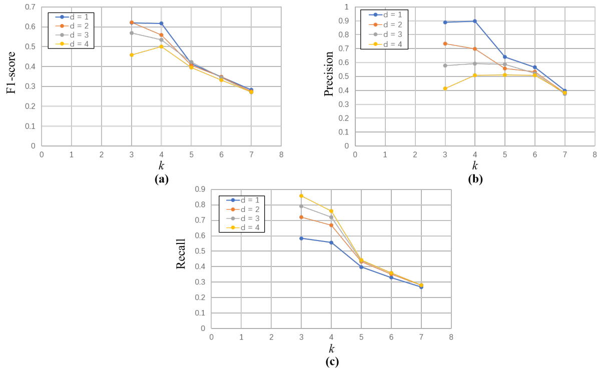

Figure 3: The F1 score(a), precision(b) and recall(c) for Truss when the query nodes are 1-hop from the ground-truth communities.

Figure 6 shows the F1 score, precision and recall for Csweep while *teleportation* varying from 0.1 to 0.45. It can be observed that for query nodes that are 1-hop from ground-truth communities, the highest F1 score is achieved when $teleportation = 0.4$. For 2-hop and 3-hop query nodes, the highest F1 score is achieved when $teleportation = 0.35$, which is smaller than the proper choice for 1-hop query nodes. This observation is consistent with the statement in Section 4 that larger teleportation for Csweep tends to find closer communities. Thus, in the following experiments, we set *teleportation* to 0.4 for 1-hop query nodes and 0.35 for 2-hop and 3-hop query nodes.

Figure 7 compares the F1 score, precision and recall of baselines and our model Csweep. It can be observed that Csweep is significantly more effective than baselines K-core and Sweep and has at least competitive effectiveness with Truss under the worst case.

5.3 Results on Robustness

As stated in Section 4, one of the most significant feature of our model is its robustness, which means that it can avoid finding isolated (or sparsely connected) attribute matched nodes that are close to the query node. In this section, we demonstrate the robustness of our model through its high resistance to *single node attack*. We will first explain the definition of *single node attack* and then show the experimental results of baselines and our model Csweep.

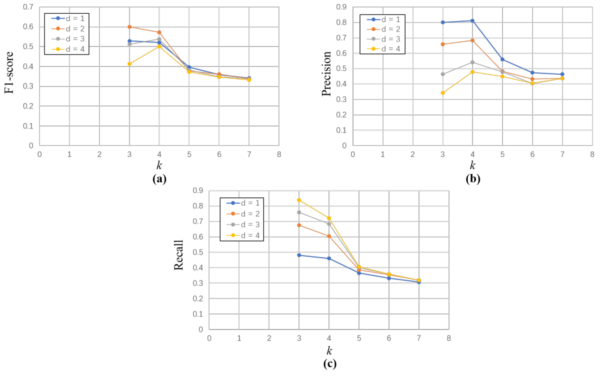

Figure 4: The F1 score(a), precision(b) and recall(c) for Truss when the query nodes are 2-hop from the ground-truth communities.

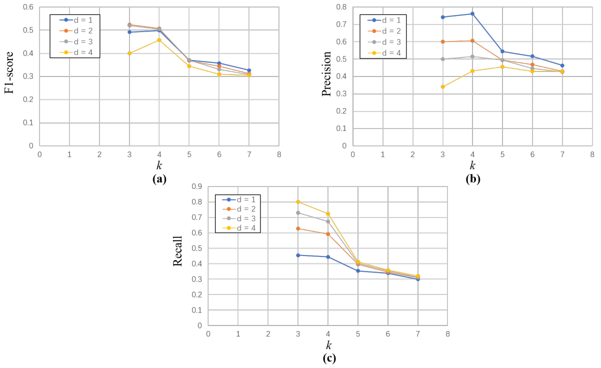

Figure 5: The F1 score(a), precision(b) and recall(c) for Truss when the query nodes are 3-hop from the ground-truth communities.

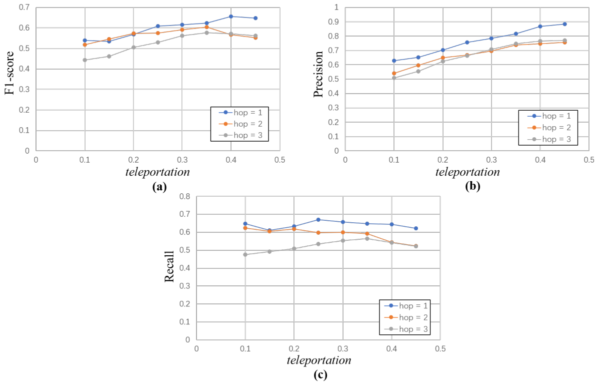

Figure 6: The F1 score(a), precision(b) and recall(c) for our model Csweep varying *teleportation*.

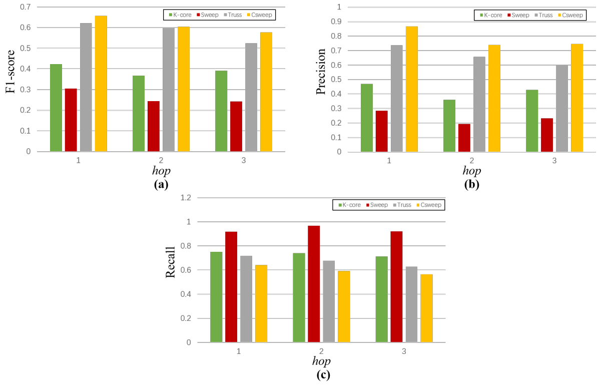

Figure 7: The F1 score(a), precision(b) and recall(c) for baselines and our model Csweep.

Definition of single node attack.

Figure 8 compares the F1 score, precision and recall of Truss before and after *single node attack*. It can be observed that all the three effectiveness metric drop sharply after *single node attack*, which means that Truss is very vulnerable to the attack. On the other hand, as illustrated in Figure 9, Sweep is much more resistant to *single node attack*. Combined with Figure 7 in the last subsection, we can derive the conclusion that Truss is very effective but vulnerable to *single node attack*; whereas Sweep is resistant to single node attack but less effective. We didn't show the results on robustness for K-core since this method is totally vulnerable to *single node attack*. It's obvious that K-core will only return the attacking node as result.

Figure 10 compares the F1 score, precision and recall of our model Csweep before and after *single node attack*. We can find that Csweep is very effective as well as resistant to *single node attack*, which combines both advantages of Truss and Sweep.

Note: It seems that the *single node attack* can be easily avoided by K-core and Truss through some adaption. By taking attribute matched nodes with top-k PPR value as seeds instead of only considering the one with largest PPR value, the attacking node will be expelled. However, this adaption has two obvious shortcomings. First, it will make these two methods less efficient since the time complexity is proportional to the number of seeds. Second, although this adaption is resistant to *single node attack*, it's still vulnerable to attacks with sparsely-connected k-nodes group. The vulnerability of K-core and Truss is their intrinsic drawback caused by the separation of closeness and cohesiveness.

5.4 Results on Efficiency

6 CONCLUSION

7 PROOFS

Let $P_G(N, p_{in}, p_{out}, p_m)$ be a planted partition model on $2N$ nodes, with 2 clusters each with exactly N nodes. We use C_0 and C_1 to denote the two clusters respectively. For each pair of nodes within the same cluster, there is an edge between them with probability p_{in}. For each pair of nodes that belong to different clusters, there is an edge between them with probability p_{out}. All the nodes

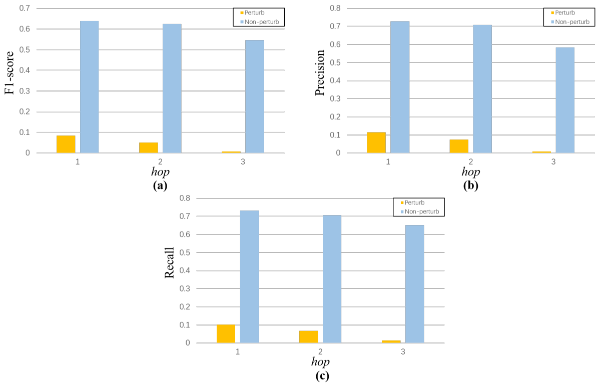

Figure 8: The comparison between F1 score(a), precision(b) and recall(c) of Truss for attacked queries and unattacked queries.

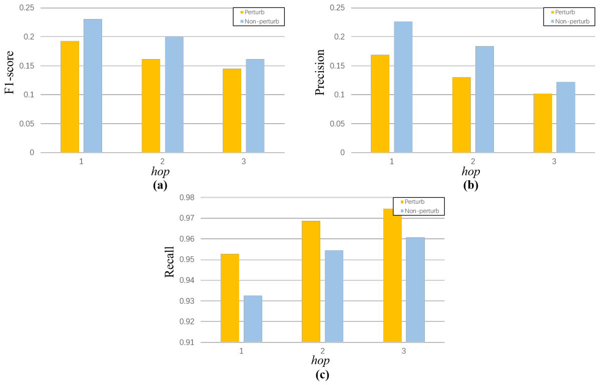

Figure 9: The comparison between F1 score(a), precision(b) and recall(c) of Sweep for attacked queries and unattacked queries.

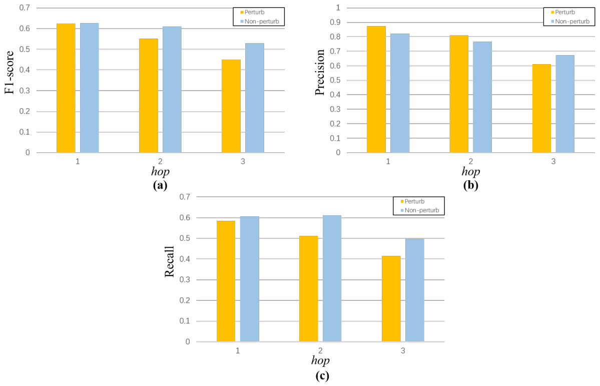

Figure 10: The comparison between F1 score(a), precision(b) and recall(c) of our model Csweep for attacked queries and unattacked queries.

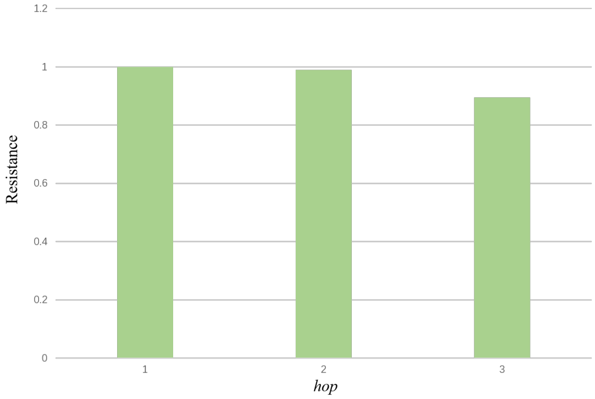

Figure 11: The resistance value of our model Csweep.

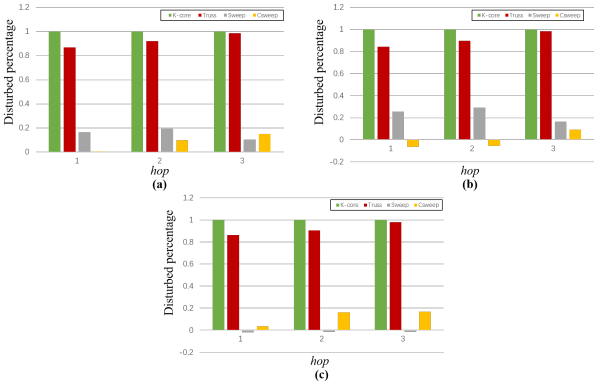

Figure 12: The attack disturbed percentage to the F1 score(a), precision(b) and recall(c) of baselines and our model.

in C_0 is attribute-matched whereas each nodes in C_1 is going to be attribute-matched with probability p_m. We use C_{1+} and C_{1-} to denote the set of attribute-matched and non-attribute-matched nodes in C_1 respectively.

Theorem 7.1. *Let G be a graph sampled from $P_G(N, p_{in}, p_{out}, p_m)$. Let $\pi(C_i)(\pi_{AS}(C_i))$ denote the average of the* personalized pagerank *value of nodes in C_i on $G(G_{AS})$ seeded in C_0. Then, for any $\delta > 0$, there exists a N sufficiently large such that*

$$\pi_{AS}(C_0) - \pi_{AS}(C_{1+}) > \pi(C_0) - \pi(C_{1+})$$

holds with probability at least $1 - \delta$.

The proof of Theorem 7.1 relies on the following two lemmas. Lemma 7.2 gives concentration bounds for the degrees of nodes in G_{AS} and G. Lemma 7.3 gives concentration bounds for the landing probabilities on G_{AS} and G.

Lemma 7.2. *Let M^{AS} and M denote the weighted adjacent matrix of G_{AS} and G, where $G \sim P_G(N, p_{in}, p_{out}, p_m)$. For any $\gamma, \delta > 0$ there exists a N sufficiently large such that*

$$\sum_{v \in C_j} M_{uv}^{AS} \in [(1-\gamma)\bar{d}_j^{AS}(u), (1+\gamma)\bar{d}_j^{AS}(u)], \forall u, \forall j \in \{0, 1+, 1-\}$$

$$\sum_{v \in C_j} M_{uv} \in [(1-\gamma)\bar{d}_j(u), (1+\gamma)\bar{d}_j(u)], \forall u, \forall j \in \{0, 1+, 1-\}$$

holds simultaneously with probability at least $1-\delta$, where node u has in expectation $\bar{d}_j^{AS}(u)(\bar{d}_j(u))$ degree to C_j on $G_{AS}(G)$ conditioned on whether it's attribute-matched.

Proof. Let's first focus on the proof on G. According to the definition of $P_G(N, p_{in}, p_{out}, p_m)$: for $u \in C_0$ we have $\bar{d}_0(u) = Np_{in}$, $\bar{d}_{1+}(u) = Np_{out}p_m$ and $\bar{d}_{1-}(u) = Np_{out}(1-p_m)$; for $u \in C_{1\pm}$, we have $\bar{d}_0(u) = Np_{out}$, $\bar{d}_{1+}(u) = Np_{in}p_m$ and $\bar{d}_{1-}(u) = Np_{in}(1-p_m)$. We observe that $\bar{d}_j(u) \propto N$ and $\sum_{v \in C_j} M_{uv}^e$ is the sum of a series of independent random variables since each edge exists independently in G and whether a node in C_1 is attribute-matched is independent from others. Thus, using standard multiplicative Chernoff bounds, for any u, any $\gamma > 0$ and any $j \in \{0, 1+, 1-\}$ we have:

$$\Pr\Big(\sum_{v \in C_j} M_{uv}^e \notin [(1-\gamma)\bar{d}_j^e(u), (1+\gamma)\bar{d}_j^e(u)]\Big) \leq O(e^{-N})$$

By union bounds, we further get:

$$\Pr\Big(\sum_{v \in C_j} M_{uv}^e \in [(1-\gamma)\bar{d}_j^e(u), (1+\gamma)\bar{d}_j^e(u)], \forall u, \forall j\Big) \geq 1 - O(Ne^{-N})$$

which clearly implies the success of the second containment.

We next turn to the proof on G_{AS}. We first give concentration bounds on the number of attribute-matched wedges in G between any pair of nodes u and v, denoted as Λ_{uv}. We can observe that the expectation $\bar{\Lambda}_{uv} = N(p_{in}^2 + p_m p_{out}^2)$ if $u, v \in C_0$; $\bar{\Lambda}_{uv} = N(p_m p_{in}^2 + p_{out}^2)$ if $u, v \in C_{1+}$; $\bar{\Lambda}_{uv} = Np_{in}p_{out}(1+p_m)$ if $u \in C_0, v \in C_{1+}$ or $u \in C_{1+}, v \in C_0$; $\bar{\Lambda}_{uv} = \Lambda_{uv} = 0$ if $u \in C_{1-}$ or $v \in C_{1-}$. Note that given u, the value of $\bar{\Lambda}_{uv}$ keeps constant while changing v within any C_j. We denote the constant as $\bar{\Lambda}_u(C_j)$.

By Chernoff bounds we have for any $\gamma_1 > 0$ and any pair of nodes u and v:

$$\Pr\Big(\Lambda_{uv} \notin [(1-\gamma_1)\bar{\Lambda}_{uv}, (1+\gamma_1)\bar{\Lambda}_{uv}]\Big) \leq O(e^{-N})$$

which further implies by union bounds that for any G:

$$\Pr\Big(\exists u, v : \Lambda_{uv} \notin [(1-\gamma_1)\bar{\Lambda}_{uv}, (1+\gamma_1)\bar{\Lambda}_{uv}]\Big) \leq O(N^2 e^{-N}) \quad (3)$$

Then, we have again by Chernoff bounds and union bounds that for any $\gamma_2 > 0$ and G:

$$\Pr\Big(\exists u : \sum_{v \in C_j} M_{uv} \notin [(1-\gamma_2)\bar{d}_j(u), (1+\gamma_2)\bar{d}_j(u)]\Big) \leq O(Ne^{-N}) \quad (4)$$

According to the definition of G_{AS}

$$\sum_{v \in C_j} M_{uv}^{AS} = \sum_{v \in C_j} \Big[(1-\alpha)M_{uv} + \alpha\Lambda_{uv} \cdot M_{uv}\Big] \quad (5)$$

Combining formula 3,4,5 and union bounds, we get

$$\sum_{v \in C_j} M_{uv}^{AS} \geq (1-\alpha)\cdot(1-\gamma_2)\bar{d}_j(u) + \alpha\cdot(1-\gamma_2)\bar{d}_j(u)\cdot(1-\gamma_1)\bar{\Lambda}_u(C_j)$$

with probability at least $1 - O(N^2 e^{-N})$. Since $\bar{d}_j^{\triangle}(u) = (1-\alpha)\bar{d}_j(u) + \alpha\bar{d}_j(u)\bar{\Lambda}_u(C_j)$ we get:

$$\sum_{v \in C_j} M_{uv}^{\triangle} \geq (1-\gamma)\bar{d}_j^{\triangle}(u) + (\gamma-\gamma_2)\bar{d}_j^{\triangle}(u) + \gamma_1(\gamma_2 - 1)\bar{d}_j(u)\bar{\Lambda}_u(C_j)$$

which implies $\sum_{v \in C_j} M_{uv}^{\triangle} \geq (1-\gamma)\bar{d}_j^{\triangle}(u)$ given:

$$\gamma \geq \gamma_1(1-\gamma_2)\frac{\bar{d}_j(u)\bar{\Lambda}_u(C_j)}{\bar{d}_j^{\triangle}(u)} + \gamma_2$$

Since γ_1 and γ_2 can be arbitrarily small, γ can also be arbitrarily small. An upper bound can be derived similarly, which leads to:

$$\Pr\Big(\sum_{v \in C_j} M_{uv}^{\triangle} \in [(1-\gamma)\bar{d}_j^{\triangle}(u), (1+\gamma)\bar{d}_j^{\triangle}(u)]\Big) \geq 1 - O(N^2 e^{-N})$$

for any u. By union bounds we further get:

$$\Pr\Big(\sum_{v \in C_j} M_{uv}^{\triangle} \in [(1-\gamma)\bar{d}_j^{\triangle}(u), (1+\gamma)\bar{d}_j^{\triangle}(u)], \forall u, \forall j\Big) \geq 1 - O(N^3 e^{-N})$$

which implies the success of the first containment. □

Note that the value of $\bar{d}_i(u)(\bar{d}_i^{AS}(u))$ keeps constant while changing u within any C_j, we denote the constant as $\bar{d}_i(C_j)(\bar{d}_i^{AS}(C_j))$. We then use $\bar{D}(C_i) = \sum_j \bar{d}_j(C_i)$ to denote the expectation volume of any node in C_i on G. Similarly, we define $\bar{D}^{AS}(C_i) = \sum_j \bar{d}_j^{AS}(C_i)$.

Lemma 7.3. *Let $R_k(C_i)(R_k^{AS}(C_i))$ denote the landing probabilities to C_i for a uniform k-step random walk on $G(G_{AS})$ seeded at C_0. For any $\epsilon, \delta > 0$, any $i \in \{0, 1+, 1-\}$ and any $K \in \mathbb{N}^+$, there is an N sufficiently large such that:*

$$R_k(C_i) \in \big[(1-\epsilon)\bar{R}_k(C_i), (1+\epsilon)\bar{R}_k(C_i)\big] \quad (6)$$

$$R_k^{AS}(C_i) \in \big[(1-\epsilon)\bar{R}_k^{AS}(C_i), (1+\epsilon)\bar{R}_k^{AS}(C_i)\big] \quad (7)$$

holds with probability at least $1-\delta$ for all $0 < k \leq K$, where $\bar{R}_k(C_i)$ and $\bar{R}_k^{AS}(C_i)$ are the solutions to the recurrence relation:

$$\bar{R}_k(C_i) = \sum_j \frac{\bar{d}_i(C_j)}{\bar{D}(C_j)}\bar{R}_{k-1}(C_j) \quad \bar{R}_k^{AS}(C_i) = \sum_j \frac{\bar{d}_i^{AS}(C_j)}{\bar{D}^{AS}(C_j)}\bar{R}_{k-1}^{AS}(C_j)$$

with $\bar{R}_0(C_0) = \bar{R}_0^{AS}(C_0) = 1, \bar{R}_0(C_{1\pm}) = \bar{R}_0^{AS}(C_{1\pm}) = 0$.

Proof. We first introduce some useful notation. Let $r_k^e(u)$ denote the landing probability to node u for a uniform k-step random walk on G_e seeded at C_0. Let $d_j^e(u) = \sum_{v \in C_j} M_{uv}^e$ denote the degree of u to C_j on G_e and $D^e(u) = \sum_v M_{uv}^e$ denote the volume of u on G_e.

We give the proof by induction. We begin with the base case, furnishing an upper bound on $R_1^e(C_i)$.

$$R_1^e(C_i) = \sum_u \frac{d_i^e(u)}{D^e(u)} r_0^e(u) = \sum_j \sum_{u \in C_j} \frac{d_i^e(u)}{D^e(u)} r_0^e(u)$$

$$\leq (\frac{1+\gamma}{1-\gamma}) \sum_j \frac{\bar{d}_i^e(C_j)}{\bar{D}_e(C_j)} \sum_{u \in C_j} r_0^e(u) \quad (8)$$

$$= (\frac{1+\gamma}{1-\gamma}) \sum_j \frac{\bar{d}_i^e(C_j)}{\bar{D}_e(C_j)} \bar{R}_0^e(C_j) = (\frac{1+\gamma}{1-\gamma}) \bar{R}_1^e(C_i) \quad (9)$$

The inequality (8) is derived directly from Lemma 7.2. The first equation of (9) is true since the random walk is seeded at C_0, which

means $\sum\limits_{u \in C_0} r_0^e(u) = 1 = \bar{R}_0^e(C_0)$ and $\sum\limits_{u \in C_{1\pm}} r_{1\pm}^e(u) = 0 = \bar{R}_0^e(C_{1\pm})$.
Next, for our induction we assume that

$$R_k^e(C_i) \le (\frac{1+\gamma}{1-\gamma})^k \cdot \bar{R}_k^e(C_i)$$

We upper-bound $R_{k+1}^e(C_i)$:

$$\begin{aligned} R_{k+1}^e(C_i) &= \sum_u \frac{d_i^e(u)}{D^e(u)} r_k^e(u) = \sum_j \sum_{u \in C_j} \frac{d_i^e(u)}{D^e(u)} r_k^e(u) \\ &\le (\frac{1+\gamma}{1-\gamma}) \sum_j \frac{\bar{d}_i^e(C_j)}{\bar{D}_e(C_j)} R_k^e(C_j) \\ &\le (\frac{1+\gamma}{1-\gamma})^{k+1} \sum_j \frac{\bar{d}_i^e(C_j)}{\bar{D}_e(C_j)} \bar{R}_k^e(C_j) = (\frac{1+\gamma}{1-\gamma})^{k+1} \cdot \bar{R}_{k+1}^e(C_j) \end{aligned}$$

Similar steps will furnish the lower bound of $R_k^e(C_i)$ and double-sided bound of $R_k^\triangle(C_i)$. As a result, we have:

$$R_k^e(C_i) \in \big[(\frac{1-\gamma}{1+\gamma})^k \cdot \bar{R}_k^e(C_i), (\frac{1+\gamma}{1-\gamma})^k \cdot \bar{R}_k^e(C_i)\big]$$

$$R_k^\triangle(C_i) \in \big[(\frac{1-\gamma}{1+\gamma})^k \cdot \bar{R}_k^\triangle(C_i), (\frac{1+\gamma}{1-\gamma})^k \cdot \bar{R}_k^\triangle(C_i)\big]$$

Formula (6) and (7) is achieved by setting:

$$\gamma \le \min\Big(\frac{1 - \sqrt[K]{1-\epsilon}}{1 + \sqrt[K]{1-\epsilon}}, \frac{\sqrt[K]{1+\epsilon} - 1}{1 + \sqrt[K]{1+\epsilon}}\Big)$$

□

We then give the proof of Theorem 7.1 based on Lemma 7.3.

Proof. We show that for any $1 \le k \le K$ and $0 < \alpha < 1$, we have $\bar{R}_k^{AS}(C_0) - \bar{R}_k(C_0) > 0$ for sufficiently large N, which will lead to the result in Theorem 7.1 naturally since $(\pi^{AS}(C_0) - \pi(C_0))$ is a linear combination of $(\bar{R}_k^{AS}(C_0) - \bar{R}_k(C_0))$ with all coefficients positive.
Let $p = \frac{p_{in}}{p_{out}} > 1$.Note that $\frac{\bar{d}_{1-}^\triangle(C_j)}{\bar{D}^\triangle(C_j)} = O(N^{-1})$ for $j \in \{0, 1+\}$, which means we can omit them given N is sufficiently large. Thus, $\bar{R}_k^\triangle(C_{1-}) = \Big(\frac{\bar{d}_{1-}^\triangle(C_{1-})}{\bar{D}^\triangle(C_{1-})}\Big)^k \bar{R}_0^\triangle(C_{1-}) = 0$. By further omitting quantities that are $O(N^{-1})$, we can derive:

$$\begin{aligned} \bar{R}_k^\triangle(C_0) &= \frac{\bar{d}_0^\triangle(C_0)}{\bar{D}^\triangle(C_0)} \bar{R}_{k-1}^\triangle(C_0) + \frac{\bar{d}_0^\triangle(C_{1+})}{\bar{D}^\triangle(C_{1+})} \bar{R}_{k-1}^\triangle(C_{1+}) \\ &= \frac{(p^2 + p_m)\bar{R}_{k-1}^\triangle(C_0)}{p^2 + 2p_m + p_m^2} + \frac{(1 + p_m)(1 - \bar{R}_{k-1}^\triangle(C_0))}{1 + 2p_m + p^2 p_m^2} \end{aligned}$$

Let $a_\triangle = \frac{(p^2+p_m)}{p^2+2p_m+p_m^2} - \frac{(1+p_m)}{1+2p_m+p^2p_m^2}$ and $b_\triangle = \frac{(1+p_m)}{1+2p_m+p^2p_m^2}$, we have:

$$\bar{R}_k^\triangle(C_0) = a_\triangle \bar{R}_{k-1}^\triangle(C_0) + b_\triangle$$

which implies:

$$\bar{R}_k^\triangle(C_0) = a_\triangle^k + \frac{b_\triangle}{1 - a_\triangle}(1 - a_\triangle^k)$$

We can observe that $\bar{R}_k^\triangle(C_0)$ is not influenced by α when N is sufficiently large.

Thus, we can get:

$$\begin{aligned} \bar{R}_k(C_0) &= \frac{\bar{d}_0(C_0)}{\bar{D}(C_0)} \bar{R}_{k-1}(C_0) + \frac{\bar{d}_0(C_{1+})}{\bar{D}(C_{1+})} \bar{R}_{k-1}(C_{1+}) \\ &= \frac{p \cdot \bar{R}_{k-1}(C_0)}{p + p_m} + \frac{(1 - \bar{R}_{k-1}(C_0))}{1 + p \cdot p_m} \end{aligned}$$

Similarly, let $a_e = \frac{p}{p+p_m} - \frac{1}{1+p \cdot p_m}$ and $b_e = \frac{1}{1+p \cdot p_m}$, we have:

$$\bar{R}_k^e(C_0) = a_e^k + \frac{b_e}{1 - a_e}(1 - a_e^k)$$

It's easy to check that $a_\triangle > a_e$ and $\frac{b_e}{1-a_e} < \frac{b_\triangle}{1-a_\triangle} < 1$. Thus, we have:

$$\begin{aligned} \bar{R}_k^e(C_0) &= a_e^k + \frac{b_e}{1 - a_e}(1 - a_e^k) \\ &< a_\triangle^k + \frac{b_e}{1 - a_e}(1 - a_\triangle^k) \\ &< a_\triangle^k + \frac{b_\triangle}{1 - a_\triangle}(1 - a_\triangle^k) = \bar{R}_k^\triangle(C_0) \end{aligned}$$

from which we can derive the conclusion directly. □

For any subgraph S of G, we use $\phi(S, G)$ to denote the conductance of S and let $S_\triangle^* = S\phi(S, G_\triangle)$ and $S_e^* = S\phi(S, G_e)$. For any two subgraphs S_0 and S_1, we use $J(S_0, S_1)$ to denote the Jaccard index between them. The following theorem shows that $S_\triangle^*$ recovers C_0 better than S_e^* when $G \sim P_G(N, L, p_{in}, p_{out}, p_m)$.

Theorem 7.4. *Let G be a graph sampled from $P_G(N, L, p_{in}, p_{out}, p_m)$. Then, for any $\delta > 0$, there exists a N sufficiently large such that*

$$J(S_\triangle^*, C_0) > J(S_e^*, C_0)$$

holds with probability at least $1 - \delta$.

Proof. □

REFERENCES

[1] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local graph partitioning using pagerank vectors. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. IEEE, 475–486.

[2] Reid Andersen and Yuval Peres. 2009. Finding sparse cuts locally using evolving sets. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*. ACM, 235–244.

[3] Nicola Barbieri, Francesco Bonchi, Edoardo Galimberti, and Francesco Gullo. 2015. Efficient and effective community search. *Data mining and knowledge discovery* 29, 5 (2015), 1406–1433.

[4] Lijun Chang, Wei Li, Lu Qin, Wenjie Zhang, and Shiyu Yang. 2017. pSCAN: Fast and Exact Structural Graph Clustering. *IEEE Transactions on Knowledge and Data Engineering* 29, 2 (2017), 387–401.

[5] Fan Chung. 2007. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences* 104, 50 (2007), 19735–19740.

[6] David Combe, Christine Largeron, Elöd Egyed-Zsigmond, and Mathias Géry. 2012. Combining relations and text in scientific network clustering. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 1248–1253.

[7] David Combe, Christine Largeron, Mathias Géry, and Előd Egyed-Zsigmond. 2015. I-louvain: An attributed graph clustering method. In *International Symposium on Intelligent Data Analysis*. Springer, 181–192.

[8] Wanyun Cui, Yanghua Xiao, Haixun Wang, Yiqi Lu, and Wei Wang. 2013. Online search of overlapping communities. In *Proceedings of the 2013 ACM SIGMOD international conference on Management of data*. ACM, 277–288.

[9] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 991–1002.

[10] TA Dang and Emmanuel Viennet. 2012. Community detection based on structural and attribute similarities. In *International conference on digital society (icds)*. 7–12.

[11] Issam Falih, Nistor Grozavu, Rushed Kanawati, and Younès Bennani. 2017. Anca: Attributed network clustering algorithm. In *International Conference on Complex Networks and their Applications*. Springer, 241–252.

[12] Issam Falih, Nistor Grozavu, Rushed Kanawati, and Younès Bennani. 2018. Community detection in attributed network. In *Companion Proceedings of the The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1299–1306.

[13] Yixiang Fang, Reynold Cheng, Yankai Chen, Siqiang Luo, and Jiafeng Hu. 2017. Effective and efficient attributed community search. *The International Journal on Very Large Data Bases* 26, 6 (2017), 803–828.

[14] Shayan Oveis Gharan and Luca Trevisan. 2012. Approximating the expansion profile and almost optimal local graph clustering. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*. IEEE, 187–196.

[15] Xin Huang and Laks VS Lakshmanan. 2017. Attribute-driven community search. *Proceedings of the VLDB Endowment* 10, 9 (2017), 949–960.

[16] Xin Huang, Laks VS Lakshmanan, Jeffrey Xu Yu, and Hong Cheng. 2015. Approximate closest community search in networks. *Proceedings of the VLDB Endowment* 9, 4 (2015), 276–287.

[17] Nasif Muslim. 2016. A combination approach to community detection in social networks by utilizing structural and attribute data. *Soc. Netw* 5, 1 (2016), 11–15.

[18] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka. 2015. SCAN++: efficient algorithm for finding clusters, hubs and outliers on large-scale graphs. *Proceedings of the VLDB Endowment* 8, 11 (2015), 1178–1189.

[19] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 939–948.

[20] Daniel A Spielman and Shang-Hua Teng. 2013. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM J. Comput.* 42, 1 (2013), 1–26.

[21] Karsten Steinhaeuser and Nitesh V Chawla. 2008. Community detection in a large real-world social network. In *Social computing, behavioral modeling, and prediction*. Springer, 168–175.

[22] Yubao Wu, Ruoming Jin, Jing Li, and Xiang Zhang. 2015. Robust local community detection: on free rider effect and its elimination. *Proceedings of the VLDB Endowment* 8, 7 (2015), 798–809.

[23] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas AJ Schweiger. 2007. Scan: a structural clustering algorithm for networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 824–833.

[24] Zhiqiang Xu, Yiping Ke, Yi Wang, Hong Cheng, and James Cheng. 2014. GBAGC: a general bayesian framework for attributed graph clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 9, 1 (2014), 5.

[25] Jaewon Yang, Julian McAuley, and Jure Leskovec. 2013. Community detection in networks with node attributes. In *2013 IEEE 13th International Conference on Data Mining*. IEEE, 1151–1156.

[26] Chen Zhe, Aixin Sun, and Xiaokui Xiao. 2019. Community Detection on Large Complex Attribute Network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2041–2049.

[27] Yang Zhou, Hong Cheng, and Jeffrey Xu Yu. 2009. Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment* 2, 1 (2009), 718–729.