Efficient Algorithms for the Identification of Top-k Structural Hole Spanners in Large Social Networks

Wenzheng Xu, *Member, IEEE*, Mojtaba Rezvani, Weifa Liang, *Senior Member, IEEE*, Jeffrey Xu Yu, *Senior Member, IEEE*, and Chengfei Liu

Abstract—Recent studies show that individuals in a social network can be divided into different groups of densely connected communities, and these individuals who bridge different communities, referred to as structural hole spanners, have great potentials to acquire resources/information from communities and thus benefit from the access. Structural hole spanners are crucial in many real applications such as community detections, diffusion controls, viral marketing, etc. In spite of their importance, little attention has been paid to them. Particularly, how to accurately characterize the structural hole spanners and how to devise efficient yet scalable algorithms to find them in a large social network are fundamental issues. In this paper, we study the top-k structural hole spanner problem. We first provide a novel model to measure the quality of structural hole spanners through exploiting the structural hole spanner properties. Due to its NP-hardness, we then devise two efficient yet scalable algorithms, by developing innovative filtering techniques that can filter out unlikely solutions as quickly as possible, while the proposed techniques are built up on fast estimations of the upper and lower bounds on the cost of an optimal solution and make use of articulation points in real social networks. We finally conduct extensive experiments to validate the effectiveness of the proposed model, and to evaluate the performance of the proposed algorithms using real world datasets. The experimental results demonstrate that the proposed model can capture the characteristics of structural hole spanners accurately, and the structural hole spanners found by the proposed algorithms are much better than those by existing algorithms in all considered social networks, while the running times of the proposed algorithms are very fast.

Index Terms—Social networks, top-k structural hole spanners, all-pairs shortest paths, filtering techniques, lower and upper bound estimations, articulation points

1 INTRODUCTION

THE last decade witnessed the exponential growth of a variety of large-scale networks such as social networks, citation networks, collaboration networks, biological networks, wireless networks, etc. With the unprecedented scale growth of network size, there are high demands for developing efficient yet scalable algorithms to explore some unique properties of such massive networks. Most social networks exhibit the so-called community structure property. That is, the vertices in a network can be grouped into different sets of cohesive groups (communities) [13], where vertices in the same community share similar attributes, interests and resources. The communities in a network play a significant role in information diffusion within the network, information within a community circulates very quickly and diffuses to other communities through community boundaries or bridges. On the other hand, there is a consensus among social scientists [10] that a person who plays a bridge role between different communities can acquire more potential resources from these communities and has more control over the information that is being transmitted. For example, Burt [10] studied social structures of many organizations and introduced the notion of *structural holes* as positions that can bridge diverse groups and bring benefits to the beholder. It is shown that the information obtained from people in the same community tends to be homogeneous, while the information through the contacts with people from different communities are much more non-redundant [27]. Therefore, a person who develops relations with people from multiple different communities will gain more benefits. *Structural hole spanners* were studied initially by Lou *et al.* [21] and later by Rezvani *et al.* [26]. For example, a community in an academic collaboration network represents a group of people with the similar research interests, and people (structural hole spanners) who can bridge different research communities are more potent to combine ideas from different research groups and create interdisciplinary works. Structural hole spanners have a wide range of practical applications. For example, in community detection, identifying central hubs that connect different groups can help isolate and identify communities [3], [34]. In Epidemic diseases and rumors spreading, quarantining structural hole spanners can stop the spread of infection and rumors into other communities [7], [15], [22]. In viral marketing, the most influential structural hole spanners can speed-up the

- *W. Xu is with College of Computer Science, Sichuan University, Chengdu, 610065, P.R. China. Email: wenzheng.xu3@gmail.com*
- *M. Rezvani and W. Liang are with Research School of Computer Science, The Australian National University, Canberra, ACT 2601, Australia. Email: mojtaba.rezvani@anu.edu.au, wliang@cs.anu.edu.au*
- *J. X. Yu is with Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong. Email: yu@se.cuhk.edu.hk*
- *C. Liu is with Department of Computer Science and Software Engineering, Swinburne University of Technology, Melbourne, VIC 3122, Australia. Email: cliu@swin.edu.au*

new product marketings to different groups [19], [35], [6], [31], [29], [23], [30]. In graph compression, structural holes are good candidates for k-shattering [18] as they connect diverse parts of a network together, and their removal will result in the network disconnected.

There are several pioneering studies on the identification of structural hole spanners in social networks [14], [20], [21], [28]. For example, Lou *et al.* [21] introduced a model for structural hole spanners, and proposed two algorithms for identifying structural hole spanners, under an assumption that communities are given in advance. However, their work relies heavily on communities while finding communities in a large social network is painstaking, and the quality of the solution delivered by their algorithm is fully determined by the found communities. If the quality of the found communities are poor, the quality of the solution consisting of structural hole spanners will be low as well. Rezvani *et al.* [26] recently studied this problem and proposed several fast heuristics for it. There are other related studies that aim to discover the structural hole spanners from a social network, using the topological structure of the network. Goyal *et al.* [14] considered each structural hole spanner as a vertex that lies on a large number of shortest paths, which is similar to the betweenness centrality. Since calculating the number of shortest paths on which each vertex lies is time-consuming, Tang *et al.* [28] formulated structural hole spanners in a network as the vertices which lie on a large number of shortest paths of length two only. These mentioned models however failed to capture the essential properties of structural hole spanners, which can be illustrated by Fig. 1. It can be seen that vertex v_1, rather than vertex v_2, lies on a large number of shortest paths of length two, while vertex v_2, instead of vertex v_1 is a better structural hole spanner as it bridges more communities. Fig. 1 demonstrates that vertex v_2 is vital for shortest paths between the vertices in different communities. In other words, upon the removal of vertex v_2, the shortest path between vertices in different communities will be obliterated.

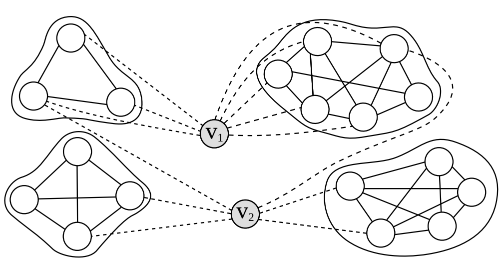


Fig. 1. An illustration of structural hole spanners, where each closed area represents a community, and vertices v_1, v_2 represent structural hole spanners that span multiple communities.

Since structural hole spanners usually bridge different communities, we have an important observation that structural hole spanners are in the shortest paths between the vertices in different communities. Their removals will result in the increase on lengths of shortest paths between these vertices. For example, vertices v_1 and v_2 in Fig. 1 play a key role in the shortest paths between the nodes in different communities and their removal can significantly increase the lengths between the other vertices, while the impact of the removal of other vertices on the shortest paths is insignificant. Based on this observation, in this paper, we propose a model based on the *mean distance* of the network [5] for modeling the structural hole spanners, which is the average of the lengths of all-pairs shortest paths in the network. We consider the structural hole spanner problem as a set of vertices whose removal will result in the maximum increase on the mean distance of vertices in the residual network, and we term *the top-k structural hole spanner problem* as the problem of finding a set of k vertices whose removal will result in the maximum increase on the mean distance. Following this definition of structural hole spanners, it can be seen that vertex v_2 is identified as a better structural hole spanner than vertex v_1 in Fig. 1, since its removal will disconnect two communities and dramatically increase the mean distance of vertices in the residual network, whereas the removal of vertex v_1 does not disconnect any communities. To the best of our knowledge, this is the first time that a novel, top-k structural hole spanner problem is formulated, its NP-hardness is proven, and efficient yet scalable algorithms are proposed. Unlike most existing studies that assumed that either all communities are given in advance or the structural hole spanner finding relies on the community detections in a network, our model relies only on the network topological structure itself.

The main contributions of this paper are summarized as follows. We first study the top-k structural hole spanner problem in social networks by formulating it as a novel optimization problem and showing its NP-hardness. We then devise two efficient yet scalable algorithms for it through the development of innovative filtering techniques that can filter out unlikely solutions as early as possible. The invented techniques are built up on fast estimations on the upper and lower bounds on the cost of an optimal solution and an observation of that the articulation points in a real social network usually are the structural hole spanners of the network. We finally validate the accuracy and effectiveness of the proposed model, and evaluate the performance of the proposed algorithms through extensive experiments, using real datasets. Experimental results demonstrate that the proposed algorithms are very promising, and the found hole spanners can connect more and larger communities in comparison with those by existing algorithms. The empirical evaluation clearly shows that the proposed algorithms outperform the other heuristics in terms of accuracy and their running times are orders of magnitude faster.

It must be mentioned that this paper is an extended version of our recent conference paper [26] with substantial new material additions. The significant differences between the conference version and this journal version are as follows. (i) A detailed proof of the NP-hardness of the top-k structural hole spanner problem is given (see Section 3 and the supplementary materials), while only a sketch proof of the NP-hardness in the conference version was provided due to the limited space. (ii) Two novel algorithms `Greedy` and `AP_Greedy` for the top-k structural hole spanner problem are proposed in Sections 4 and 5, respectively, by developing innovative filtering techniques to filter out unlikely solutions as early as possible. The filtering techniques are based on fast estimations of the upper and lower bounds on the cost of an optimal solution and the exploration of articulation points in a social network jointly. (iii) To evaluate the performance of the proposed algorithms, extensive experiments have been conducted in Section 6, and experimental results show that the structural hole spanners found by the proposed algorithms are much better than those found by existing algorithms including the ones in our conference paper for all considered social networks. In addition, the newly proposed algorithm `AP_Greedy` is highly scalable, which takes only about 145 seconds for finding top-50 structural hole spanners in a social network with over five million vertices and more than 27 million edges in a typical desktop machine.

The rest of the paper is organized as follows. Section 2 introduces basic notations and the problem definition. Section 3 proves the NP-hardness of the problem. Sections 4 and 5 proposes efficient algorithms for the problem. Section 6 evaluates the performance of the proposed algorithms, using real social network datasets. Section 7 reviews related works on structural hole spanners, and Section 8 concludes the paper.

2 Preliminaries and Definitions

2.1 Network Model

A social network can be modeled as an undirected graph $G = (V, E)$, where V is the set of vertices representing individuals and E is the set of edges representing the relationships between individuals. Let $n = |V|$ and $m = |E|$. A vertex in G is an *articulation point* (or *cut vertex*) of G if its removal will disconnect a connected component in G.

As G is an unweighted graph, assume that each edge has a weight of 1, and we term each edge $e \in E$ as a *real edge*. The distance d^G_{uv} between two vertices u and v in G is the length of the shortest path between them. We assume that $d^G_{vv} = 0$ for any vertex $v \in V$. Given a subset V_S of V, let $G[V \setminus V_S]$ be the induced subgraph of G by the vertices in $V \setminus V_S$. We abbreviate $G[V \setminus V_S]$ by $G \setminus V_S$ if no ambiguities arise. The *average distance* $c(v)$ [5] from a vertex v in G to the other vertices then is

$$c(v) = \frac{\sum_{u \in V} d^G_{uv}}{n-1}. \tag{1}$$

The mean distance of a graph G is defined as follows.

$$c(G) = \frac{\sum_{v \in V} c(v)}{|V|} = \frac{\sum_{v \in V}\sum_{u \in V} d^G_{uv}}{(n-1)|V|} = \frac{\sum_{v \in V}\sum_{u \in V} d^G_{uv}}{n(n-1)}. \tag{2}$$

The sum of lengths of all pairs shortest paths in G [26] is

$$C(G) = \sum_{u \in V}\sum_{v \in V} d^G_{uv} = n(n-1)c(G). \tag{3}$$

Note that if G is disconnected, to make the mean distance of G still be valid, the distance between two vertices not in the same connected component is defined by a sufficiently large value ζ to avoid the infinite distance. This value should be larger than the sum of lengths of all-pairs shortest paths in any connected component of G, e.g., $\zeta = n^3/3$, as the upper bound on the sum of lengths of all pairs shortest paths in a n-vertex graph is no more than $n^3/3$ [25].

Lemma 1. [25] Let $G = (V, E)$ be an unweighted connected graph with $n = |V|$ vertices, then the sum of lengths of all-pairs shortest paths in G is no more than $n^3/3$.

2.2 Problem Definition

Given a social network $G = (V, E)$ and a positive integer k, the *top-k structural hole (SH) spanner problem* in G is to find a subset of vertices V_S ($V_S \subset V$) with $|V_S| = k$, such that the removal of the vertices in V_S from G will result in the maximum increase on the sum of all-pairs shortest distances in the induced subgraph $G \setminus V_S$ (also see [26]), i.e., the objective is to maximize the value of $C(G \setminus V_S) - C(G)$,

$$\max_{V_S \subset V,\ |V_S| = k} \{C(G \setminus V_S) - C(G)\}, \tag{4}$$

which is equivalent to

$$\max_{V_S \subset V,\ |V_S| = k} \{C(G \setminus V_S)\}. \tag{5}$$

Communities in a network G are defined as the groups of vertices with dense connections internally and sparse connections externally [13]. Due to node density, the distance between two vertices in the same community is small and the removal of any member from the same community does not change the distance between the other members considerably. In contrast, structural hole spanners bridge different communities, and their removal can dramatically increase the distance between the vertices in these communities [10]. The removal of top-k structural hole spanners from a network will result in the maximum number of communities disconnected in comparison with other k-vertex removals, thereby significantly increasing the mean distance of the nodes in the remaining network. Fig. 1 illustrates the impact of the removal of structural hole spanners on the mean distance. Specifically, the proposed model and the defined problem capture three important characteristics of structural hole spanners as follows [26].

(i) Given an individual u who bridges multiple communities and another individual v who contacts with people only in his/her community, individual u is considered by the model to be a better structural hole spanner than individual v, since the connections among individuals within the communities to which individual v belongs are strong, and the absence of v only slightly increases the distance among the other individuals in the network. In contrast, individual u connects people who are in different communities, its absence can dramatically increase the distance between them, as they are loosely connected.

(ii) Given an individual u who bridges large communities and another individual v who bridges small communities, individual u is considered to be a better structural hole spanner than individual v, since its removal will disconnect more people in the network.

(iii) Given an individual u who bridges many communities and another individual v who bridges only a few, individual u is considered to be a better structural hole spanner, since its removal will increase the distance between more communities (even if they are smaller).

Note that the proposed model relies only on the network topological structure itself, which is more generic and intuitively captures the properties of the structural hole spanners in real social networks in a unified way.

3 NP-hardness

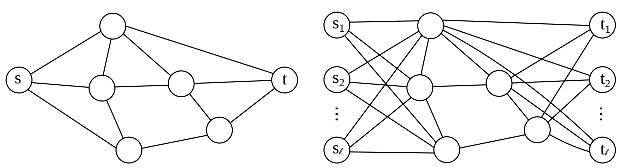
In this section we show that the top-k structural hole spanner problem is NP-hard, by a non-trivial reduction from a known NP-hard problem - the *Most Vital Node Problem* (MVNP) [4], which is defined as follows.

Given an undirected graph $G = (V \cup \{s,t\}, E)$, a pair of nodes s and t, and a positive integer k, assume that there is no edge between vertices s and t in G and their vertex connectivity $\kappa^G(s,t)$ is no less than $k+1$, the MVNP is to find a subset V_S of V with $|V_S| = k$ such that the length of the shortest path between s and t in subgraph $G[(V \setminus V_S) \cup \{s,t\}]$ of G is maximized.

Theorem 1. The top-k structural hole spanner problem in a social network $G = (V, E)$ is NP-hard.

Proof: Given an instance of the MVNP in an undirected graph $G = (V \cup \{s,t\}, E)$ with $n = |V \cup \{s,t\}|$, a pair of vertices s and t in G, and a positive integer k, an instance of the top-k structural hole spanner problem in another undirected graph $G' = (V \cup S \cup T, E')$ can be constructed as follows.

Let $l = 4n^6$. Sets of vertices S and T are obtained by duplicating vertices s and t with each l times, i.e., $S = \{s_1, s_2, \ldots, s_l\}$ and $T = \{t_1, t_2, \ldots, t_l\}$. For any two different vertices $u, v \in V$, an edge (u,v) is added to E' if there is an edge $(u,v) \in E$. For each vertex $v \in V$, l edges $(v, s_1), (v, s_2), \ldots,$ and (v, s_l) (or $(v, t_1), (v, t_2), \ldots,$ and (v, t_l)) are added to E' if edge (v, s) (or (v,t)) is contained in E. The construction of G' is illustrated in Fig. 2. Clearly, it can be verified that $\kappa^{G'}(s_i, t_j) = \kappa^G(s,t)$ for any vertex $s_i \in S$ and any vertex $t_j \in T$ with $1 \le i, j \le l$, and $d^{G'}_{uv} = d^G_{uv}$ for every pair of vertices u and v.



(a) G (b) G'

Fig. 2. G' is constructed from G by replicating vertices s and t and their incident edges l times.

The MVNP in $G = (V \cup \{s,t\}, E)$ can be solved by a reduction to the SH problem in G' as follows.

We first show that an optimal solution to the problem in G' does not contain any vertex s_i or t_i, thus, it is a feasible solution for the MVNP. We then prove that an optimal solution to the structural hole spanner problem is indeed an optimal solution to the MVNP.

Assume that set $V_S \subset V \cup S \cup T$ is an optimal solution to the top-k structural hole spanner problem in G', i.e., $C(G' \setminus V_S) = \max_{V'_S \subset V \cup S \cup T, |V'_S| = k}\{C(G' \setminus V'_S)\}$. Following Lemma 3 in the supplementary materials, set V_S does not contain any vertices in S or T. Therefore, V_S is a feasible solution to the MVNP. We show that V_S is an optimal solution for the MVNP in G in the following. Let

$$x = (C(G' \setminus V_S) - 4l(n_V - k) - 4l(l-1) - (n_V - k)(n_V - k - 1))/2l^2, \quad (6)$$

We distinguish the proof into two cases: (i) $d^{G \setminus V_S}_{st} = \lfloor x \rfloor$; and (ii) $\lfloor x \rfloor = \max_{V'_S \subset V, |V'_S| = k}\{d^{G\setminus V'_S}_{st}\} = d^{G \setminus V_S}_{st}$. Set V_S then is an optimal solution to the MVNP in G.

We first consider Case (i). $d^{G \setminus V_S}_{st} = \lfloor x \rfloor$. We can see that $d^{G \setminus V_S}_{st} = d^{G' \setminus V_S}_{s_i t_j}$ for any pair of vertices $s_i \in S$ and $t_j \in T$. To this end, we show that $d^{G' \setminus V_S}_{s_i t_j} \le \lfloor x \rfloor$ and $d^{G' \setminus V_S}_{s_i t_j} \ge \lfloor x \rfloor$.

Following Lemma 4 in the supplementary materials, we have that $C(G' \setminus V_S) \ge 4l(n_V - k) + 4l(l-1) + (n_V - k)(n_V - k - 1) + 2l^2 d^{G' \setminus V_S}_{s_i t_j}$, thus $d^{G' \setminus V_S}_{s_i t_j} \le x$. Since the value of $d^{G' \setminus V_S}_{s_i t_j}$ is a positive integer, $d^{G' \setminus V_S}_{s_i t_j} \le \lfloor x \rfloor$.

We then show that $d^{G' \setminus V_S}_{s_i t_j} \ge \lfloor x \rfloor$ by contradiction.

Assume $d^{G' \setminus V_S}_{s_i t_j} < \lfloor x \rfloor$, then $d^{G' \setminus V_S}_{s_i t_j} \le \lfloor x \rfloor - 1$. Following Lemma 4 in the supplementary materials, we have

$$\begin{aligned} & C(G' \setminus V_S) \\ \le\ & 4l(n_V - k)\zeta + 4l(l-1) + (n_V - k)(n_V - k - 1)\zeta + 2l^2 d^{G' \setminus V_S}_{s_j t_j}, \end{aligned}$$

since we assumed that $d^{G' \setminus V_S}_{s_i t_j} \le \lfloor x \rfloor - 1$,

$$\begin{aligned} & C(G' \setminus V_S) \\ \le\ & 4l(n_V - k)\zeta + 4l(l-1) + (n_V - k)(n_V - k - 1)\zeta + 2l^2(\lfloor x \rfloor - 1) \\ \le\ & 4l(n_V - k)\zeta + 4l(l-1) + (n_V - k)(n_V - k - 1)\zeta + 2l^2(x - 1), \end{aligned}$$

we then simply substitute x by its value from Eq.(6),

$$\begin{aligned} & C(G' \setminus V_S) \\ \le\ & 4l(n_V - k)\zeta + 4l(l-1) - 2l^2 + (n_V - k)(n_V - k - 1)\zeta \\ & + C(G' \setminus V_S) - 4l(n_V - k) - 4l(l-1) - (n_V - k)(n_V - k - 1) \\ <\ & C(G' \setminus V_S) + 4l(n_V - k)\zeta + (n_V - k)(n_V - k - 1)\zeta - 2l^2, \end{aligned}$$

since $n_V - k - 1 < n_V - k < n$ and $\zeta = n^3$. Therefore,

$$C(G' \setminus V_S) \quad < \quad C(G' \setminus V_S) + 4ln^4 + n^5 - 2l^2.$$

Since $l = 4n^6$, we have

$$C(G' \setminus V_S) < C(G' \setminus V_S) + 16n^{10} + n^5 - 32n^{12} < C(G' \setminus V_S), \quad (7)$$

where inequality (7) results in a contradiction. Therefore, $d^{G' \setminus V_S}_{s_j t_j} \ge \lfloor x \rfloor$. We thus have $d^{G' \setminus V_S}_{s_j t_j} = \lfloor x \rfloor$.

The rest is to show Case (ii). $\lfloor x \rfloor = \max_{V'_S \subset V, |V'_S| = k}\{d^{G \setminus V'_S}_{st}\} = d^{G \setminus V_S}_{st}$. Denote by V^*_S the optimal solution to the MVNP in G, i.e., $d^{G \setminus V^*_S}_{st} = \max_{V'_S \subset V, |V'_S| = k}\{d^{G \setminus V'_S}_{st}\}$. It can be seen that $d^{G \setminus V_S}_{st} = d^{G' \setminus V_S}_{s_i t_j}$ and $d^{G \setminus V^*_S}_{st} = d^{G' \setminus V^*_S}_{s_i t_j}$ for any pair of vertices $s_i \in S$ and $t_j \in T$. Assume that $d^{G' \setminus V^*_S}_{s_i t_j} > d^{G' \setminus V_S}_{s_i t_j}$, then $d^{G' \setminus V^*_S}_{s_i t_j} \ge d^{G' \setminus V_S}_{s_i t_j} + 1 = \lfloor x \rfloor + 1 > x$. Following Lemma 4, we have

$$\begin{aligned} & C(G' \setminus V^*_S) \\ \ge\ & 4l(n_V - k) + 4l(l-1) + 2l^2 d^{G' \setminus V^*_S}_{s_j t_j} + (n_V - k)(n_V - k - 1) \\ >\ & 4l(n_V - k) + 4l(l-1) + 2l^2 \cdot x + (n_V - k)(n_V - k - 1) \\ =\ & C(G' \setminus V_S), \end{aligned} \quad (8)$$

i.e., $C(G' \setminus V^*_S) > C(G' \setminus V_S)$, which contradicts the assumption that V_S is an optimal solution. Thus, set V_S is an optimal solution to the MVNP in G. The top-k structural hole spanner problem is NP-hard, too. □

4 An Efficient Algorithm

In this section, we propose a novel greedy algorithm for the top-k structural hole spanner problem. We first introduce the basic idea behind the algorithm, and then elaborate the algorithm in detail.

4.1 Basic idea

The proposed algorithm proceeds iteratively. Within each iteration, one structural hole spanner is found. To identify a vertex v_i (i.e., a structural hole spanner) within iteration i so that the average shortest distance of the vertices in the residual network after the removal of the first i found vertices $v_1, v_2, \ldots, v_i$ is maximized. The key technique to this algorithm is the development of an innovative filtering technique to filter out unlikely solutions as early as possible. The k found vertices then form the top-k structural hole spanners in the social network. In the following we detail this greedy algorithm.

4.2 A greedy algorithm

Given a social network $G = (V, E)$ and a positive integer k, the greedy algorithm finds a set V_S of k spanners within k iterations and removes one vertex from network G within each iteration. Let $v_1, v_2, \ldots, v_k$ be the k spanners found by the greedy algorithm in the 1st, 2nd, …, and kth iteration, respectively. Denote by G_i the residual network after the removal of the first i spanners from G, i.e., $G_i(V_i, E_i) = G[V \setminus \{v_1, v_2, \ldots, v_i\}]$, $1 \leq i \leq k$. We can see G_{i-1} is the network just before the ith iteration, assuming that $G_0 = G$.

The greedy algorithm proceeds as follows. Assume that the algorithm has found the first $(i-1)$ spanners $v_1, v_2, \ldots, v_{i-1}$ prior to iteration i with $1 \leq i \leq k$. Recall that $G_{i-1}(V_{i-1}, E_{i-1}) = G[V \setminus \{v_1, v_2, \ldots, v_{i-1}\}]$, where $G[X]$ is an induced subgraph of G by the vertices in set X with $X \subseteq V$. Within iteration i, the greedy algorithm identifies a vertex v_i from graph G_{i-1} (i.e., $v_i \in V_{i-1} = V \setminus \{v_1, v_2, \ldots, v_{i-1}\}$) so that the average distance of graph $G_{i-1}[V_{i-1} \setminus \{v_i\}]$ is maximized, i.e., $v_i = \arg\max_{v_j \in V_{i-1}} \{\frac{C(G_{i-1} \setminus \{v_j\})}{n_i(n_i - 1)}\} = \arg\max_{v_j \in V_{i-1}} \{C(G_{i-1} \setminus \{v_j\})\}$ by Eq. (3), where $n_i = |V_{i-1}| - 1 = |V| - (i-1) - 1 = n - i$.

It can be seen that a simple way to find vertex v_i is to calculate the sum of all-pairs shortest distances $C(G_{i-1} \setminus \{v_j\})$ in graph $G_{i-1} \setminus \{v_j\}$ for every vertex $v_j \in V_{i-1}$. It thus takes $O(n_{i-1} n_i m_{ij}) = O(n^2 m)$ time to find vertex v_i, where the sum of all-pairs shortest distances $C(G_{i-1} \setminus \{v_j\})$ in graph $G_{i-1} \setminus \{v_j\}$ can be calculated, by running a single-source shortest paths algorithm n_i times, once for each vertex as the source in the graph, and the most efficient single-source shortest paths algorithm in an unweighted graph is to perform a breadth-first search from the source vertex [12], $n_{i-1} = n - (i-1)$, $n_i = n - i$, and m_{ij} is the number of edges in graph $G_{i-1} \setminus \{v_j\}$. In the following we propose an efficient algorithm for finding vertex v_i by reducing its finding time significantly. This is implemented through the development of an innovative filtering technique to filter out unlikely solutions as early as possible. To this end, we propose two strategies to significantly reduce the running time of finding vertex v_i.

The first one is a pruning technique, which efficiently estimates the upper bound $UB(G_{i-1} \setminus \{v_j\})$ on the cost $C(G_{i-1} \setminus \{v_j\})$, and prunes the costly calculation of $C(G_{i-1} \setminus \{v_j\})$ if the upper bound $UB(G_{i-1} \setminus \{v_j\})$ is no greater than the maximum cost among the first $(j-1)$ graphs $G_{i-1} \setminus \{v_1\}, G_{i-1} \setminus \{v_2\}, \ldots, G_{i-1} \setminus \{v_{j-1}\}$.

The second one is to fast calculate the value of $C(G_{i-1} \setminus \{v_j\})$ with tight statistical guarantees, by applying a randomized algorithm for this purpose.

The detailed algorithm for the top-k structural hole spanner problem is depicted in `Algorithm` 1.

Algorithm 1 Greedy Algorithm

Input: A network $G = (V, E)$ and a positive integer k
Output: The set V_S of top-k structural hole spanners in network G

Let $V_S \leftarrow \emptyset$ and $G_0 \leftarrow G$;
for $i \leftarrow 1$ to k **do**
  Sort vertices in graph G_{i-1} in non-increasing order of their vertex degrees, and let $v_1, v_2, \ldots, v_{n-(i-1)}$ be the sorted vertices;
  $C^0_{\max} \leftarrow 0$; /* the maximum sum of all-pairs shortest distances among the first $(j-1)$ networks */
  for $j \leftarrow 1$ to $n - (i-1)$ **do**
    Estimate the upper bound $UB(G_{i-1} \setminus \{v_j\})$ on the cost $C(G_{i-1} \setminus \{v_j\})$ of network $G_{i-1} \setminus \{v_j\}$, by invoking `Procedure` 1;
    if $UB(G_{i-1} \setminus \{v_j\}) \leq C^{j-1}_{\max}$ **then**
      $C^j_{\max} \leftarrow C^{j-1}_{\max}$;
    else
      Calculate $C(G_{i-1} \setminus \{v_j\})$ by invoking `Procedure` 2;
      if $C(G_{i-1} \setminus \{v_j\}) > C^{j-1}_{\max}$ **then**
        $v_i \leftarrow v_j$ and $C^j_{\max} \leftarrow C(G_{i-1} \setminus \{v_j\})$;
      else
        $C^j_{\max} \leftarrow C^{j-1}_{\max}$;
      end if
    end if
  end for
  Add vertex v_i to set V_S;
  Let G_i be the residual network after the removal of vertex v_i and its incident edges from G_{i-1};
end for
return Set V_S.

The pruning technique is described as follows. We sort vertices in graph G_{i-1} in non-increasing order of their degrees. The intuition behind is that it is more likely that the removal of a large degree vertex, rather than a small degree vertex, can significantly increase the shortest distances of other pairs of vertices. Let $v_1, v_2, \ldots, v_{n-(i-1)}$ be the sorted vertices in G_{i-1} in non-increasing order, and let $C^{j-1}_{\max}$ be the maximum sum of all-pairs shortest distances among the first $(j-1)$ graphs $G_{i-1} \setminus \{v_1\}, G_{i-1} \setminus \{v_2\}, \ldots, G_{i-1} \setminus \{v_{j-1}\}$, i.e., $C^{j-1}_{\max} = \max_{l=1}^{j-1}\{C(G_{i-1} \setminus \{v_l\})\}$. The pruning technique first estimates the upper bound $UB(G_{i-1} \setminus \{v_j\})$ on $C(G_{i-1} \setminus \{v_j\})$, it prunes the costly calculation of $C(G_{i-1} \setminus \{v_j\})$ if $UB(G_{i-1} \setminus \{v_j\}) \leq C^{j-1}_{\max}$. Then, $C^j_{\max} = \max\{C^{j-1}_{\max},\ C(G_{i-1} \setminus \{v_j\})\} = C^{j-1}_{\max}$.

The upper bound $UB(G_{i-1} \setminus \{v_j\})$ on $C(G_{i-1} \setminus \{v_j\})$ is estimated as follows. Assume that graph $G_{i-1} \setminus \{v_j\}$ consists of p_{ij} connected components $G'_1, G'_2, \ldots, G'_{p_{ij}}$, where p_{ij} is a positive integer. Denote by n'_l the number of vertices in connected component G'_l. Then, $\sum_{l=1}^{p_{ij}} n'_l = n_i = n - i$. Specifically, we first find a spanning tree T_l in each connected component G'_l with $1 \leq l \leq p_{ij}$. Then, for any two vertices u and v in G'_l, the shortest distance between them in tree T_l is no less than that in G'_l, and the sum of all-pairs shortest distances $C(T_l)$ in tree T_l is an upper bound on $C(G'_l)$. The upper bound $UB(G_{i-1} \setminus \{v_j\})$ on $C(G_{i-1} \setminus \{v_j\})$

thus is

$$UB(G_{i-1} \setminus \{v_j\}) = \sum_{l=1}^{p_{ij}} C(T_l) + \sum_{l=1}^{p_{ij}} n'_l(n_i - n'_l) \cdot \zeta, \quad (9)$$

where $n'_l(n_i - n'_l)$ is the number of pairs between the vertices in G'_l and the vertices not in G'_l, and ζ is the large value assigned as the shortest distance of two unreachable vertices.

The calculation of the sum of all-pairs shortest distances $C(T_l)$ in tree T_l proceeds as follows. For each edge (u, v) in tree T_l of G'_l, the removal of edge (u, v) from tree T_l will disconnect the tree into two subtrees $T_{l,u}$ and $T_{l,v}$ that contain vertices u and v, respectively. Let $n'_{l,u}$ and $n'_{l,v}$ be the numbers of vertices in the two subtrees, respectively, and $n'_{l,u} + n'_{l,v} = n'_l$. Since T_l is a tree, it can be seen that edge (u, v) is contained in $2 \cdot n'_{l,u} \cdot n'_{l,v}$ pairs of shortest paths among all-pairs shortest paths in tree T_l. Therefore, $C(T_l) = \sum_{(u,v)\in T_l} 2n'_{l,u}n'_{l,v}$ since the length of each edge is one.

`Procedure` 1 details the estimation of the upper bound $UB(G')$ on the cost $C(G')$ of graph G'.

Procedure 1 Estimate an upper bound on the sum of all-pairs shortest distances

---

Input: an unweighted network $G' = (V', E')$
Output: An upper bound $UB(G')$ on the cost $C(G')$ of graph G'

```
Assume that there are p connected components in G', and denote by G'_1, G'_2, ..., G'_p the p connected components;
Let UB(G') ← 0;    /* the upper bound */
for l ← 1 to p do
    Find a spanning tree T_l of connected component G'_l, by performing a depth-first search on G'_l, since the weight of each edge is one;
    Choose an arbitrary vertex r_l in tree T_l as its root;
    Calculate the number of vertices n'_v in the subtree rooted at each vertex v of tree T_l by performing a depth-first search starting from root r_l;
    for each edge (u, v) in tree T_l do
        Assume vertex u is the parent of vertex v in tree T_l;
        UB(G') ← UB(G') + 2n'_v(n'_l − n'_v), where n'_l is the number of vertices in tree T_l;
    end for
end for
for l ← 1 to p do
    UB(G') ← UB(G') + n'_l(n' − n'_l) · ζ, where n'_l and n' are the numbers of vertices in connected components G'_l and G', respectively;
end for
return UB(G').
```

In case the estimated upper bound $UB(G_{i-1} \setminus \{v_j\})$ on the cost $C(G_{i-1} \setminus \{v_j\})$ is greater than the maximum cost $C^{j-1}_{\max}$ among the first $(j-1)$ graphs (i.e., this filter cannot filter out vertex v_j), we then calculate the cost $C(G_{i-1} \setminus \{v_j\})$ in graph $G_{i-1} \setminus \{v_j\}$, by the randomized algorithm in [11], which estimates the sum of all-pairs shortest distances in a connected graph with a low time complexity $O(\epsilon^{-2}(n_i + m_i))$ yet within a multiplicative relative error ϵ, where ϵ is a given small constant with $0 < \epsilon \leq 1$. It must be mentioned that the randomized algorithm in [11] is only applicable to a connected graph, which may not be directly applicable to our case where $G_{i-1} \setminus \{v_j\}$ may be disconnected. Recall that graph $G_{i-1} \setminus \{v_j\}$ consists of p_{ij} connected components $G'_1, G'_2, \ldots, G'_{p_{ij}}$ with each G'_l containing n'_l vertices. The sum of all-pairs shortest distances $C(G'_l)$ in G'_l can be estimated within a multiplicative relative error ϵ, by applying the randomized algorithm in [11]. The sum of all-pairs shortest distances $C(G_{i-1} \setminus \{v_j\})$ in graph $G_{i-1} \setminus \{v_j\}$ thus is

$$C(G_{i-1} \setminus \{v_j\}) = \sum_{l=1}^{p_{ij}} C(G'_l) + \sum_{l=1}^{p_{ij}} n'_l(n_i - n'_l) \cdot \zeta. \quad (10)$$

`Procedure` 2 depicts the detailed randomized algorithm for calculating the sum of all-pairs shortest distances in a graph $G' = (V', E')$.

Procedure 2 Calculate the sum of all-pairs shortest distances in a graph

---

Input: an undirected graph $G' = (V', E')$
Output: The sum of all-pairs shortest distances $C(G')$ of G'

```
Assume there are p connected components in G' and denote by G'_1, G'_2, ..., G'_p the p connected components;
Let C(G') ← 0;
for l ← 1 to p do
    Estimate the sum of all-pairs shortest distances C(G'_l) among the connected component G_l, by applying the randomized algorithm in [11];
    C(G') ← C(G') + C(G'_l) + n'_l(n' − n'_l) · ζ, where n'_l and n' are the numbers of vertices in connected component G'_l and G', respectively;
end for
return C(G').
```

4.3 Analysis of algorithm complexity

We now analyze the time complexity of `Algorithm` 1 by the following theorem.

***Theorem* 2.** Given a social network $G = (V, E)$ and a positive integer k, there is an algorithm, `Algorithm` 1, for the top-k structural hole spanner problem in G, which takes $O(\epsilon^{-2}kn(n + m))$ time, where $n = |V|$, $m = |E|$, and ϵ is a constant with $0 < \epsilon \leq 1$.

Proof: Following `Algorithm` 1, it finds a set V_S of k structural hole spanners in G within k iterations. Within the ith iteration $(1 \leq i \leq k)$, it either estimates an upper bound $UB(G_{i-1} \setminus \{v_j\})$ on the sum of all-pairs shortest distances $C(G_{i-1} \setminus \{v_j\})$ in graph $G_{i-1} \setminus \{v_j\}$ by invoking `Procedure` 1, or calculates the cost $C(G_{i-1} \setminus \{v_j\})$ by invoking `Procedure` 2. We can see that the time complexity of `Procedure` 1 is $O(n + m)$.

The rest is to analyze the time complexity of `Procedure` 2. Recall that $G_{i-1} \setminus \{v_j\}$ consists of p_{ij} connected components $G'_1, G'_2, \ldots, G'_{p_{ij}}$. It takes $O(\epsilon^{-2}(n'_l + m'_l))$ time to estimate the sum of all-pairs shortest distances $C(G'_l)$ in G'_l, by invoking the randomized algorithm in [11], where n'_l and m'_l are the numbers of vertices and edges in G'_l, respectively. Therefore, it takes $\sum_{l=1}^{p_{ij}} O(\epsilon^{-2}(n'_l + m'_l)) = O(\epsilon^{-2}(\sum_{l=1}^{p_{ij}} n'_l + \sum_{l=1}^{p_{ij}} m'_l)) = O(\epsilon^{-2}(n + m))$ to estimate the cost of $C(G_{i-1} \setminus \{v_j\})$. The time complexity of `Algorithm` 1 thus is $O(kn\epsilon^{-2}(n + m))$. □

5 A Fast yet Scalable Algorithm

So far, we have devised a heuristic algorithm for the top-k structural hole spanner problem. However, following Theorem 2, the time complexity $O(\epsilon^{-2}kn(n+m))$ of `Algorithm` 1 is still high, especially when the network size is very large, e.g., millions of vertices. In the following we devise another fast yet scalable algorithm by making use of articulation points (APs) of network G to further prune unlikely solutions as quickly as possible. We term this algorithm as the AP-based algorithm, whose running time is only $O(k(n+m))$. In fact, its running time is linear for most real social networks, which can be seen in later experiments.

5.1 The basic idea of the proposed algorithm

One of the instinct properties of structural hole spanners in most real social networks is their tendency to connect multiple isolated communities. Such structural hole spanners are referred to the *articulation points* in graph theory. Thus, a top-k structural hole spanner usually is an articulation point of a network too. However, the number of articulation points in a real social network may be quite large, e.g., there are hundreds even thousands of articulation points in each real social network of Table 1 in Section 6.1. Efficiently identifying top-k structural hole spanners from all articulation points in a large social network is a challenging issue. In the following we devise a fast yet scalable algorithm for the top-k structural hole spanner problem, by making use of articulation points along with the filtering technique introduced in the previous section to further prune unlikely structural hole spanner candidates as early as possible.

Similar to `Algorithm` 1 proposed in the previous section, the AP-based algorithm identifies the k structural hole spanners within k iterations and removes all found spanners from network G within each iteration. Assume that the algorithm has already found the first $(i-1)$ structural hole spanners $v_1, v_2, \ldots, v_{i-1}$ in its first $(i-1)$ iterations with $1 \leq i \leq k$. Recall that $G_{i-1}(V_{i-1}, E_{i-1}) = G[V_{i-1}]$ and $V_{i-1} = V \setminus \{v_1, v_2, \ldots, v_{i-1}\}$. Within the ith iteration, it finds a vertex $v_i \in V_{i-1}$ so that the sum of all-pairs shortest distances in graph $G_{i-1} \setminus \{v_i\}$ is maximized, i.e., $v_i = arg\max_{v_j \in V_{i-1}}\{C(G_{i-1} \setminus \{v_j\})\}$.

Unlike that the calculation of the cost $C(G_{i-1} \setminus \{v_j\})$ of each graph $G_{i-1} \setminus \{v_j\}$ with $v_j \in V_{i-1}$ in `Algorithm` 1 is very time-consuming, the AP-based algorithm efficiently finds the ith structural hole spanner v_i by exploiting only the articulation points in G_{i-1}, since it is very likely that the removal of an articulation point from graph G_{i-1} will significantly increase the shortest distances of other vertices in the residual network.

5.2 A fast yet scalable algorithm

We partition the vertices in graph $G_{i-1}(V_{i-1}, E_{i-1})$ into two disjoint sets: the set of articulation points V_{AP} and the set of non-articulation points V_{NAP} $(= V_{i-1} \setminus V_{AP})$, where a vertex $v_j \in V_{i-1}$ is contained in set V_{AP} if it is an *articulation point (AP)* in G_{i-1} (i.e., the removal of vertex v_j from G_{i-1} disconnects a connected component in G_{i-1}); otherwise, vertex v_j is contained in set V_{NAP}. We now estimate an upper bound UB_j on the sum of all-pairs shortest distances $C(G_{i-1} \setminus \{v_j\})$ in graph $G_{i-1} \setminus \{v_j\}$ for each non-articulation point (non-AP) $v_j \in V_{NAP}$, and a lower bound LB_l on $C(G_{i-1} \setminus \{v_l\})$ in graph $G_{i-1} \setminus \{v_l\}$ for each AP $v_l \in V_{AP}$. It can be seen that an AP v_l is a better structural hole spanner than a non-AP v_j if $LB_l > UB_j$. We will show that the ith structural hole spanner v_i is an AP in set V_{AP} with the maximum lower bound if its lower bound is greater than the maximum upper bound among non-APs, i.e., $v_i = \arg\max_{v_l \in V_{AP}}\{LB_l\}$ if $\max_{v_l \in V_{AP}}\{LB_l\} > \max_{v_j \in V_{NAP}}\{UB_j\}$.

We first estimate the upper bound UB_j on $C(G_{i-1} \setminus \{v_j\})$ for each non-AP $v_j \in V_{NAP}$. Assume that G_{i-1} consists of p_i connected components $G'_1, G'_2, \ldots, G'_{p_i}$ with each G'_t containing n'_t vertices, $1 \leq t \leq p_i$. Then, $\sum_{t=1}^{p_i} n'_t = n_{i-1} = n - (i-1)$. Similar to Eq. (10), the sum of all-pairs shortest distances in G_{i-1} is $C(G_{i-1}) = \sum_{t=1}^{p_i} C(G'_t) + \sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t) \cdot \zeta$. For a non-AP $v_j \in V_{NAP}$, assume that vertex v_j is contained in connected component G'_s of G_{i-1}. Since vertex v_j is not an articulation point, its removal will not disconnect G'_s. Let $G''_s = G'_s \setminus \{v_j\}$. Then, G''_s is still connected. The sum of all-pairs shortest distances in $G_{i-1} \setminus \{v_j\}$ thus is

$$\begin{aligned}
&C(G_{i-1} \setminus \{v_j\}) \\
&= \sum_{t=1, t\neq s}^{p_i} C(G'_t) + C(G''_s) + \sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t)\zeta - 2(n_{i-1} - n'_s)\zeta \quad (11) \\
&\leq \sum_{t=1, t\neq s}^{p_i} {n'}_t^3/3 + {n'}_s^3/3 + \sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t)\zeta - 2(n_{i-1} - n'_s)\zeta \quad (12) \\
&\leq n^3/3 + \sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t) \cdot \zeta - 2(n_{i-1} - n'_s)\zeta, \quad (13) \\
&\leq (\sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t) + 2n'_s + 1 - 2n_{i-1}) \cdot \zeta, \text{ as } n^3/3 \leq \zeta \\
&= UB_j, \quad (14)
\end{aligned}$$

where in Eq. (11), $2(n_{i-1} - n'_s)$ is the number of unreachable pairs of vertices between vertex v_j and the vertices not in connected component G'_s, and v_j is contained in connected component G'_s, Ineq. (12) holds by Lemma 1, and Ineq. (12) holds since $\sum_{t=1, t\neq s}^{p_i} n'_t + n'_s = n - i \leq n$. Note that the upper bound UB_j can be easily calculated, by counting the number of unreachable pairs of vertices in $G_{i-1} \setminus \{v_j\}$, i.e., $\sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t) + 2n'_s - 2n_{i-1}$.

We then estimate the lower bound LB_l on $C(G_{i-1} \setminus \{v_l\})$ for each AP $v_l \in V_{AP}$. Assume that vertex v_l is contained in a connected component G'_s. Since v_l is an articulation point, the removal of v_l will disconnect G'_s into, say, p_{sl} connected components $G''_1, G''_2, \ldots, G''_{p_{sl}}$. Denote by n''_t the number of vertices in G''_t. Then, $\sum_{t=1}^{p_{sl}} n''_t = n'_s - 1$. The sum of all-pairs shortest distances in $G_{i-1} \setminus \{v_l\}$ thus is

$$\begin{aligned}
&C(G_{i-1} \setminus \{v_l\}) \\
&= \sum_{t=1, t\neq s}^{p_i} C(G'_t) + \sum_{t=1}^{p_{sl}} C(G''_t) + \sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t)\zeta \\
&\qquad -2(n_{i-1} - n'_s)\zeta + \sum_{t=1}^{p_{sl}} n''_t(n'_s - 1 - n''_t)\zeta \\
&\geq (\sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t) + \sum_{t=1}^{p_{sl}} n''_t(n'_s - 1 - n''_t) + 2n'_s - 2n_{i-1})\zeta \\
&= LB_l, \quad (15)
\end{aligned}$$

where $\sum_{t=1}^{p_{sl}} n''_t(n'_s - 1 - n''_t)$ is the number of unreachable pairs of vertices in the p_{sl} components. Notice that $\sum_{t=1}^{p_i} n'_t(n_{i-1} - n'_t) + \sum_{t=1}^{p_{sl}} n''_t(n'_s - 1 - n''_t) + 2n'_s - 2n_{i-1}$ is the number of unreachable pairs of vertices in $G_{i-1} \setminus \{v_l\}$. Also, it can be seen that the value of lower bound LB_l is maximized when the size n'_s of connected component G'_s is large and all the p_{sl} connected components have roughly equal sizes, i.e., $n''_1 \approx n''_2 \approx \cdots \approx n''_{p_{sl}}$.

We finally propose the fast yet scalable algorithm, by making use of both the lower bound estimations on APs and the upper bound estimations on non-APs to filter out unlikely solutions as quickly as possible. Specifically, within iteration i with $1 \le i \le k$, the AP-based algorithm first finds the set V_{AP} in $G_{i-1}(V_{i-1}, E_{i-1})$ and calculates the lower bound LB_l on the cost $C(G_{i-1} \setminus \{v_l\})$ of $G_{i-1} \setminus \{v_l\}$ for each AP $v_l \in V_{AP}$, by invoking `Procedure` 3. It then chooses a vertex in set V_{AP} with the maximum lower bound as the ith structural hole spanner v_i if its lower bound is no less than the maximum upper bound among all non-APs, otherwise, $(\max_{v_l \in V_{AP}}\{LB_l\} \le \max_{v_j \in V_{i-1} \setminus V_{AP}}\{UB_j\})$, the AP-based algorithm finds a vertex v_i by applying the method from Line 4 to Line 16 in `Algorithm` 1. The correctness of the choice of v_i is guaranteed by Lemma 2, which will be shown later.

The detailed algorithm is described in `Algorithm` 2.

Algorithm 2 `AP_Greedy`

Input: A network $G = (V, E)$ and a positive integer k
Output: The set V_S of top-k structural hole spanners in G

Let $V_S \leftarrow \emptyset$ and $G_0 \leftarrow G$;
for $i \leftarrow 1$ to k **do**
  Find the set V_{AP} in graph $G_{i-1}(V_{i-1}, E_{i-1})$ and calculate the lower bound LB_l on the cost $C(G_{i-1} \setminus \{v_l\})$ of $G_{i-1} \setminus \{v_l\}$ for each $v_l \in V_{AP}$, by invoking `Procedure` 3;
  Compute the upper bound UB_j on the cost $C(G_{i-1} \setminus \{v_j\})$ of graph $G_{i-1} \setminus \{v_j\}$ for each vertex $v_j \in V_{i-1} \setminus V_{AP}$, by Eq. (14);
  if $\max_{v_l \in V_{AP}}\{LB_l\} > \max_{v_j \in V_{i-1} \setminus V_{AP}}\{UB_j\}$ **then**
    $v_i \leftarrow \arg\max_{v_l \in V_{AP}}\{LB_l\}$;
  else
    Find the ith structural hole spanner v_i in G_{i-1}, by invoking a similar method from Line 4 to Line 16 in `Algorithm` 1;
  end if
  $V_S \leftarrow V_S \cup \{v_i\}$;
  Let G_i be the residual network after the removal of vertex v_i and its incident edges from G_{i-1};
end for
return Set V_S.

5.3 Finding all articulation points in graph G_{i-1}

We now describe how to find all articulation points and how to estimate the lower bound LB_l for each $v_l \in V_{AP}$ efficiently, using only one Depth-First Search (DFS) traversal on G_{i-1}.

Let v be an articulation point of G_{i-1}. In the DFS tree construction starting from an arbitrary root $r \in V_{i-1}$, assume that $u_1, u_2, \ldots, u_p$ are the children of vertex v in the DFS tree. Let V'_i be the set of vertices in the subtree T_i rooted at u_i and G'_i the connected component of G_{i-1} induced by the vertices in V'_i with $1 \le i \le p$. Let G'_0 be the connected component containing the ancestors of v in the DFS tree. Following the DFS traversal property, all edges in G_{i-1} can be partitioned into two categories: the "tree edges" and the "non-tree edges", respectively. And all non-tree edges are "back edges", which means that one endpoint of each of such edges is a descendant of another endpoint (or another endpoint is a proper ancestor of the endpoint) in the DFS tree. Clearly, there is no edges between any two connected components G'_i and G'_j with $i \neq j$ and $1 \le i, j \le p$, by the DFS traversal property. If there is a back edge between a vertex in G'_i and a vertex in G'_0, then both G'_i and G'_0 are in the same connected component when the removal of v from G_{i-1}. An illustration of this case is shown in Fig 3.

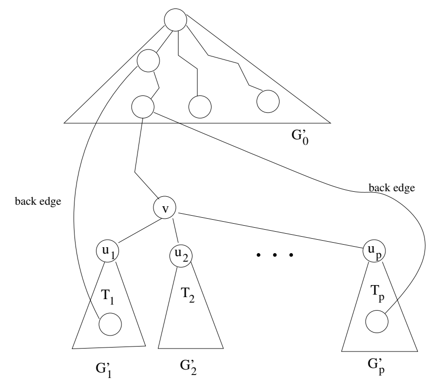


Fig. 3. An illustration of exploring an articulation point v and its p children $u_1, u_2, \ldots, u_p$ during a DFS traversal on G_{i-1}.

Assume that p' connected components among the p components derived from the p children of v have back edges. Then, the removal of v will result in $(p - p' + 1)$ connected components. For the sake of convenience, we assume that these $(p - p' + 1)$ connected components are $G''_1, G''_2, \ldots, G''_{p-p'+1}$ with each containing n''_t vertices. The lower bound LB_v of vertex v then is

$$LB_v = \sum_{t=1}^{p-p'+1} |n''_t| \cdot (n_{i-1} - 1 - |n''_t|) \cdot \zeta. \qquad (16)$$

The linear-time procedure `Procedure` 3 of detecting all articulation points and the calculation of the lower bound of each articulation point is detailed as follows.

A vertex v is identified as an articulation point of G_{i-1} if a subtree rooted at one of its children does not contain any back edges. The induced subgraph by the set of vertices in this subtree is a connected component after the removal of vertex v from G_{i-1}. The number of vertices contained in each such connected component is the number of descendants of that child in the DFS tree. To keep track of the number of descendants of each vertex when performing the DFS traversal on G_{i-1} and to identify those children of the vertex without any back edges, the lower bound LB_v of v (as an articulation point) can be easily calculated. The detailed implementation of this is given in `Procedure` 3 and `Procedure` 4, respectively.

5.4 Complexity analysis of the proposed algorithm

Lemma 2. Given a graph $G_{i-1} = (V_{i-1}, E_{i-1})$, let V_{AP} be the set of articulation points in G_{i-1} and $V_{NAP} = V_{i-1} \setminus V_{AP}$. Assume the maximum lower bound among APs is greater than the maximum upper bound among non-APs, i.e., $\max_{v_l \in V_{AP}}\{LB_l\} > \max_{v_j \in V_{NAP}}\{UB_j\}$. Then, the removal of vertex v_i will result in the maximum cost $C(G_{i-1} \setminus \{v_i\})$ among vertices in V_{i-1}, where v_i is an

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2651825, IEEE Transactions on Knowledge and Data Engineering

Procedure 3 Finding articulation points and their lower bound calculations

Find the number n'_t of vertices in each connected component G'_t of graph G_{i-1};
for each vertex $u \in V_{i-1}$ **do**
  $u.child \leftarrow 0$; /* number of children of u in the DFS tree*/
  $u.visited \leftarrow$ '$false$'; /* whether u was visited before */
  $u.AP \leftarrow$ '$false$'; /* whether u is an articulation point */
  $LB_u \leftarrow 0$; /* the lower bound of vertex u */
end for
$time \leftarrow 0$;
for each vertex $u \in G_{i-1}$ **do**
  if $u.visted =$ '$false$' **then**
    invoke Procedure **Modified-DFS**(G_{i-1}, u);
  end if
end for

Procedure 4 Modified-DFS(G_{i-1}, u)

$u.visited \leftarrow$ '$true$';
$time \leftarrow time + 1$;
$u.discovered \leftarrow time$; /*the discovery time of vertex u*/
$u.lowest \leftarrow time$; /* the smallest discovery time of any neighbor of u's descendants (through a back-edge) */
$cc_0 \leftarrow n'_t - 1$, where u is contained in connected component G'_t, and n'_t is the number of vertices in G'_t; /* the number of vertices in G'_0 after removing vertex u */
$u.descendant \leftarrow 0$; /* the number of descendants of u in the DFS tree */
for each edge (u, v) in graph G_{i-1} **do**
  if $v.visited =$ '$false$' **then**
    $v.\pi \leftarrow u$; /* u is the parent of v */;
    $u.child \leftarrow u.child + 1$;
    invoke Procedure **Modified-DFS**(G_{i-1}, v);
    $u.descendant \leftarrow u.descendant + v.descendant$;
    $u.lowest \leftarrow \min(u.lowest, v.lowest)$;
    if ($v.lowest \geq u.discovered$) OR ($u$ is the root AND $u.child > 1$) **then**
      $u.AP \leftarrow$ '$true$'; /* v is disconnected without u */
      $LB_u \leftarrow LB_u + v.descendant \times (n_{i-1} - 1 - v.descendant)$;
      $cc_0 \leftarrow cc_0 - v.descendant$; /* the subtree of v is not part of G'_0 */;
    end if
  else if $v \neq u.\pi$ **then**
    $u.lowest \leftarrow \min(u.lowest, v.discovered)$;
  end if
end for
$u.descendant \leftarrow u.descendant + 1$; /*including itself*/
if $u.AP =$ '$true$' **then**
  $LB_u \leftarrow LB_u + cc_0 \times (n_{i-1} - 1 - cc_0)$.
end if

AP in set V_{AP} with the maximum lower bound, i.e., $v_i = \arg\max_{v_l \in V_{AP}}\{LB_l\}$.

Proof: By the assumption that $\max_{v_l \in V_{AP}}\{LB_l\} > \max_{v_j \in V_{NAP}}\{UB_j\}$, we know the ith structural hole spanner must be an articulation point. For any vertex $v_l \in V_{AP} \setminus \{v_i\}$, we show that $C(G_{i-1} \setminus \{v_i\}) \geq C(G_{i-1} \setminus \{v_l\})$. Without loss of generality, assume the lower bounds of different APs are different. Since v_i is the vertex with the maximum lower bound, $LB_i > LB_l$. In fact, $LB_i \geq LB_l + \zeta$, since the value of a lower bound is the product of an integer and ζ, by Eq. (15). On the other hand, the value of $LB_l + \zeta$ is an upper bound on $C(G_{i-1} \setminus \{v_l\})$ by Eq. (14). Therefore,

$$C(G_{i-1} \setminus \{v_i\}) \geq LB_i \geq LB_l + \zeta = UB_l \geq C(G_{i-1} \setminus \{v_l\}). \quad (17)$$

The lemma then follows. □

Theorem 3. Given a social network $G = (V, E)$ and a positive integer k, let n_{AP} be the number of articulation points in G. There is a fast yet scalable algorithm, Algorithm 2, for the top-k structural hole spanner problem, which takes time $O(k(m+n))$ if n_{AP} is no less than k; otherwise ($n_{AP} < k$), Algorithm 2 takes $O(n_{AP}(n+m) + (k - n_{AP})\epsilon^{-2}n(n+m))$ time, where $n = |V|$, $m = |E|$, and ϵ is a constant with $0 < \epsilon \leq 1$.

Proof: Following Algorithm 2, the detection of all articulation points in G_{i-1} and the calculation of their lower bounds take $O(n+m)$ time by Procedure 3. For each vertex u, its adjacency list is traversed exactly once and the number of descendants and children are calculated in the post-traversal in DFS. The total amount of time for calculating the number of descendants of each vertex in the DFS tree is $O(n)$. Also, it takes $O(m+n)$ time to compute the upper bound UB_j of non-APs in G_{i-1} by Eq. (14). The ith SH spanner v_i thus is an AP of G_{i-1} if the number of articulation points in it is no less than k, since the maximum lower bound among APs usually is much larger than the maximum upper bound among non-APs. Since the number of articulation points in a real social network usually is quite large, e.g., there are hundreds even thousands of articulation points in each network of Table 1, the time complexity of Algorithm 2 is $O(k(n+m))$ in practice. □

6 Performance Evaluation

In this section we evaluate the performance of the proposed algorithms using different datasets. We start with the experimental environment settings, then investigate the effectiveness of the proposed model of structural hole spanners, compared with other models using real datasets. We finally study the performance of the proposed algorithms, using the datasets in Table 1.

6.1 Experimental environment setting

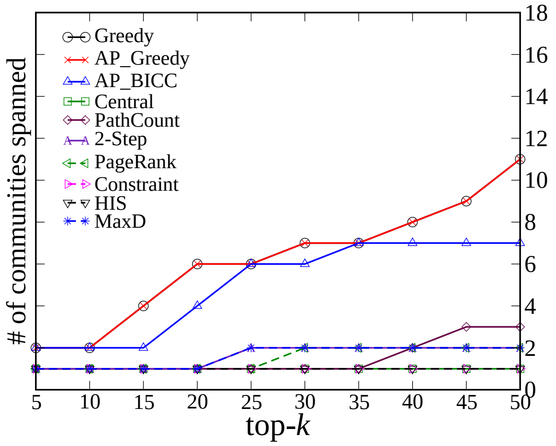
We adopt five real-world datasets, which are listed in Table 1, where GR-QC is the collaboration network from arXiv[1], covering collaborations between authors of papers submitted to General Relativity and Quantum Cosmology category. Epinions is an online social network of a general consumer review site Epinions[2]. The Twitter dataset was obtained from [21]. The DBLP-2011 dataset is the collaboration network obtained from the DBLP web site[3], and the LiveJournal dataset describes the social network of free on-line blogging community[4].

TABLE 1
Statistics of four real-world networks

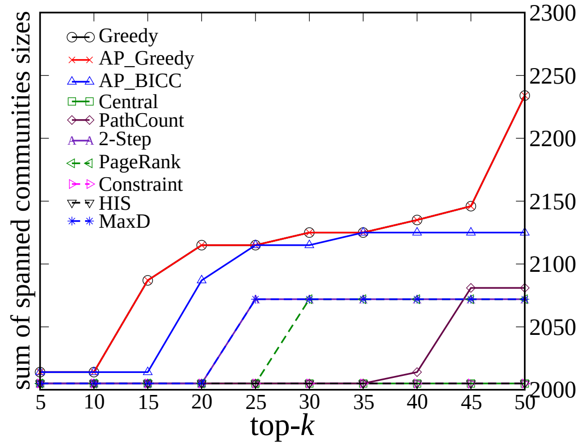
Dataset	$\lvert V\rvert$	$\lvert E\rvert$	# of APs (%)
GR-QC	5,244	14,484	813 (15.5%)
Epinions	75,887	221,176	10,746 (14.1%)
Twitter	92,180	188,971	12,549 (13.6%)
DBLP-2011	986,324	3,353,618	90,607 (9.2%)
LiveJournal	5,363,260	27,440,444	616,973 (11.5%)

1. http://arxiv.org/
2. http://epinions.com/
3. http://www.informatik.uni-trier.de/∼ley/db/
4. http://livejournal.com/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2651825, IEEE Transactions on Knowledge and Data Engineering



(a) the number of communities connected to identified structural hole spanners



(b) the sum of sizes of the communities connected to found spanners

Fig. 4. Performance of different algorithms in network `Coauthor`.

To evaluate the performance of the proposed algorithms `Greedy` and `AP_Greedy` in this paper, we will compare the following nine state-of-the-art algorithms.

(1) Algorithm `AP_BICC` chooses the top-k structural hole spanners based on articulation points and the bounded inverse closeness centrality, which was proposed in [26].

(2) Algorithm `Central` finds k vertices with the smallest average shortest distances to other vertices [26], [11], i.e., the k vertices with the maximum closeness centrality. The fastest randomized algorithm proposed in [11] is applied to compute the average shortest distances to other vertices for every vertex.

(3) Algorithm `PathCount` [14] is similar to betweenness centrality and assigns each vertex a score that is the number of shortest paths among all-pairs shortest paths, on which the vertex lies, then selects the top-k vertices with the highest scores.

(4) Algorithm `2-Step` [28] assigns each vertex a score that is the number of pairs of its neighbors without edges between them, then selects the top-k highest scores.

(5) Algorithm `PageRank` [24] assigns each vertex v a PageRank score $r(v)$ that is the visiting probability of v by a random surfer, $r(v) = 1/n$ initially. The algorithm then updates $r(v)$ with a new value $r(v) = (1-\alpha)/n + \alpha \sum_{(u,v)\in E} r(u)/\deg(u)$, where $\alpha = 0.85$ is the random jump parameter. It finally chooses the top-k vertices with the highest PageRank scores.

(6) Algorithm `Constraint` [10] measures the constraint on each vertex by its neighbors and selects k vertices with the lowest constraint scores.

(7) Algorithm `HAM` [16] proposed a harmonic modularity method to tackle the detections of both communities and structural hole spanners simultaneously. Notice that we compared algorithm `HAM` only under the smallest network `GR-QC` with 5,244 nodes, due to that the space complexity of that algorithm is very high, i.e., $O(n^2)$.

(8) Algorithm `HIS` [21] assigns each vertex v a score that simulates the likelihood of v as a structural hole spanner across the given subset of communities, assuming that l communities are given.

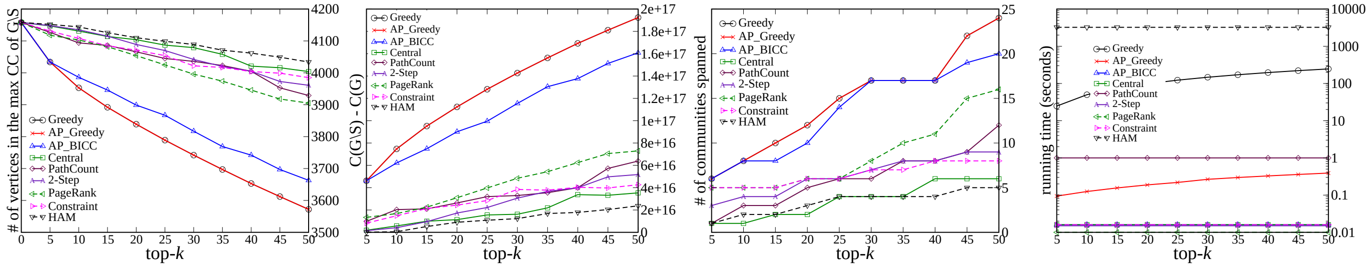
(9) Algorithm `MaxD` [21] is to find a set of k vertices such that the minimum cut of communities will be reduced significantly, after removing these vertices, assuming that l communities are given. For any pair of communities, the algorithm selects $\lceil 2k/(l(l-1)) \rceil$ vertices as structural hole spanners using a greedy strategy. In each round, it chooses the vertex whose removal will result in a maximum decrease of the minimum cut.

Notice that all our experiments were performed on a desktop with Intel(R) Core(TM) i7-4790 CPU (3.6 GHz), 8 GB RAM, and the operating system of Windows 8.1 enterprise.

6.2 Effectiveness of the proposed model

We first evaluate the effectiveness of the proposed model using the definition in [10], such as the number of communities that an individual spans and the size of each spanned community. Given a social network $G = (V, E)$ and the set of communities $\mathcal{C} = \{C_1, C_2, \ldots, C_l\}$ in G, suppose S is the set of structural hole spanners found by an algorithm, where C_i is a community in G and $1 \le i \le l$. The quality of the solution S is measured by two metrics: (1) the number of distinct communities that individuals in S span; and (2) the sum of the sizes of communities spanned by S. We evaluate the performance of different algorithms using these two metrics in order to find out the extent to which our model maps the real structural hole spanners. We use the `Coauthor` dataset containing 53,442 vertices and 127,968 edges, which has been used for the same purpose in [21]. The communities in this network are publication venues, e.g, journals or conferences, and authors who published in the same journal or conference form a community.

Fig. 4 shows the performance of different algorithms by increasing the number of structural hole spanners k from 1 to 50, using the `Coauthor` dataset. Fig. 4 (a) clearly illustrates that algorithms `Greedy` and `AP_Greedy` outperform the other algorithms. For example, the numbers of communities connected to structural hole spanners found by algorithms `Greedy` and `AP_Greedy` are as high as eleven when $k = 50$, while the number by algorithm `AP_BICC` is only seven and the numbers by the rest of the algorithms are no more than three. On the other hand, Fig. 4 (b) shows that the number of users in the communities connected to the structural hole spanners delivered by algorithms `Greedy` and `AP_Greedy` are much more than those by the other mentioned algorithms. Fig. 4 validates our claim in Section 2.2 that our model can identify the vertices connecting



(a) numbers of vertices in the maximum CC of graph $G \setminus S$.

(b) the value of $C(G \setminus S) - C(G)$

(c) numbers of communities connected to found SH spanners

(d) running time

Fig. 5. Performance of different algorithms in network GR-QC.

with larger communities. In a nutshell, algorithms Greedy and AP_Greedy can guarantee to find structural hole spanners connecting more communities and larger communities in this dataset while the other algorithms find individuals connecting less numbers of communities.

6.3 Performance on real datasets

We now evaluate the performance of the mentioned algorithms including the proposed algorithms Greedy and AP_Greedy, and benchmark structure-based algorithms AP_BICC, Central, PathCount, 2-Step, PageRank, Constraint, and HAM in different social networks listed in Table 1. In this experiment, we do not compare community-based algorithms such as HIS and MaxD, since no ground-truth communities are available and using any type of community detection method can be subject to unfairness.

Fig. 5 (a) plots the number of vertices in the maximum connected component in graph $G \setminus S$ by different algorithms in network GR-QC, where S is the set of structural hole spanners found by an algorithm, and $G \setminus S$ is the graph by removing vertices in S and their incident edges from graph G. Intuitively, good structural hole spanners are connected to many yet large communities and their removal from G will disconnect these communities. As a result, the size of the maximum connected component in graph $G \setminus S$ significantly decreases. In contrast, the removal of poor structural hole spanners may not disconnect the communities to which they are connected, or disconnect only a few yet small communities from large communities. Then, the size of the maximum connected component in graph $G \setminus S$ is still large. We thus can see that the smaller the size of the maximum connected component in graph $G \setminus S$ is, the better the found spanners are. Fig. 5 (a) shows that the sizes of the maximum connected components in graph $G \setminus S$ by algorithms Greedy and AP_Greedy are much smaller than those by the other algorithms. For example, the sizes of the maximum connected components by algorithms Greedy and AP_Greedy decrease from 4,158 vertices in graph G to only 3,572 vertices in graph $G \setminus S$ when $k = 50$, while the size by algorithm AP_BICC is 3,663 vertices, and the sizes by the other algorithms are even no less than 3,900 vertices. Therefore, it can be seen that algorithms Greedy and AP_Greedy find much better structural hole spanners than the other algorithms in network GR-QC.

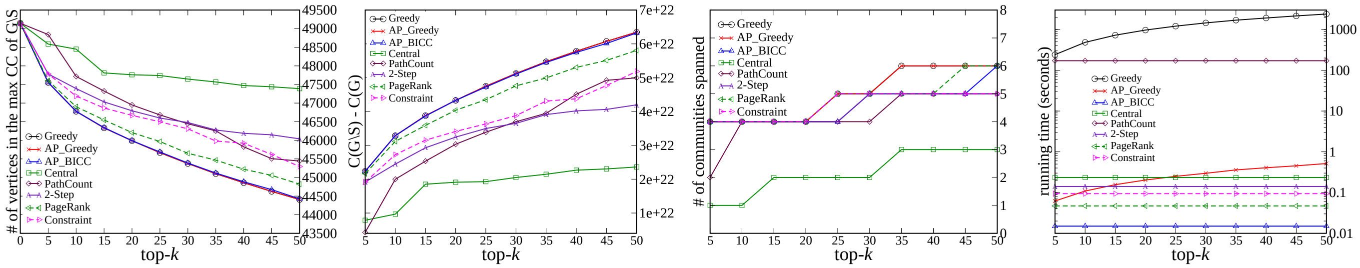
Fig. 5 (b) investigates the performance of different algorithms in terms of the optimization objective in network GR-QC, which was shown in Eq. (4) of Section 2.2, i.e., $C(G \setminus S) - C(G)$. It can be seen that the values of $C(G \setminus S) - C(G)$ in the solutions delivered by algorithms Greedy and AP_Greedy are much larger than those by the other algorithms. Specifically, the values of $C(G \setminus S) - C(G)$ by algorithms Greedy and AP_Greedy are 20% larger than that by algorithm AP_BICC when $k = 50$, and even from 2.5 times to 14 times the values by algorithms Central, PathCount, 2-Step, PageRank, Constraint, and HAM. The rationale behind is explained as follows. On one hand, although the structural hole spanners identified by algorithm AP_BICC connect different communities, some structural hole spanners may connect to the same communities. On the other hand, the individuals found by the other algorithms may not connect many communities, though these algorithms find important individuals in the network, such as algorithm PageRank detects the individuals with high reputation, algorithms Central, and PathCount find vertices by degree centrality, closeness centrality, and betweenness centrality, respectively, and algorithm 2-Step identifies the vertices with the smallest clustering coefficients.

Fig. 5 (c) plots the number of communities that individuals in S span. Since ground-truth communities are unavailable, we adopt the notion of *conductance* to define communities in a social network [36]. It is validated that conductance is one of the two definitions that consistently give the best performance in identifying ground-truth communities [36]. Based on the conductance definition, we find communities in a social network by applying the approximate PageRank algorithm in [2]. Fig. 5 (c) demonstrates that the numbers of communities connected to the spanners delivered by the proposed algorithms Greedy and AP_Greedy are larger than those by the other algorithms.

Fig. 5 (d) studies the running times of different algorithms in network GR-QC. It can be seen that the running times of algorithms HAM and Greedy are much longer than that of the other algorithms. For example, the running times of these two algorithms are 3,232 seconds and 246 seconds, respectively, when $k = 50$, while the running times of the other algorithms are no more than one second, where the time complexities of algorithms HAM, Greedy, PathCount, Central, AP_Greedy, 2-Step, Constraint, PageRank, and AP_BICC are $O(cn^3)$, $O(\epsilon^2 n(n+m))$, $O(n(n+m))$, $O(\epsilon^2(n+m))$, $O(k(n+m))$, $O(d_{\max}(n+m))$, $O(d_{\max}(n+m))$, $O(l(n+m))$, and $O(n \log n + m)$, respectively, where $d_{\max}$ is the maximum vertex degree, c and l are the numbers of iterations needed so that algorithms HAM and PageRank converge, respectively.

We then study the performance of different algorithms

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2651825, IEEE Transactions on Knowledge and Data Engineering



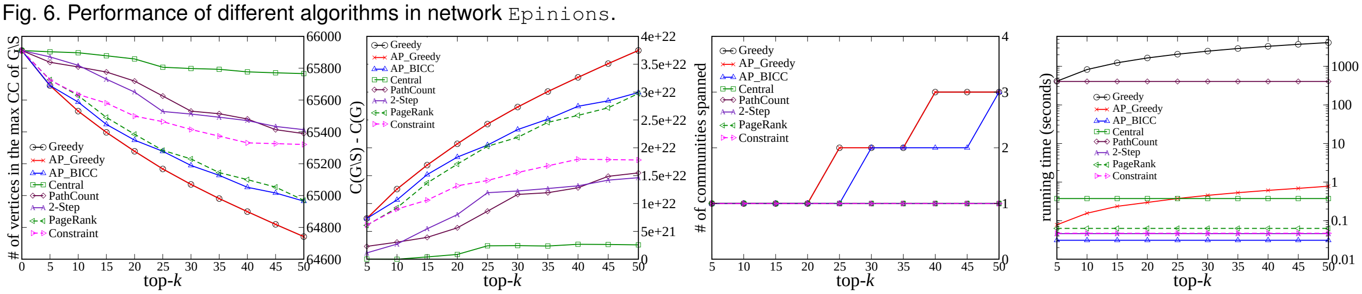
(a) numbers of vertices in the maximum CC of graph $G \setminus S$.

(b) the value of $C(G \setminus S) - C(G)$

(c) numbers of communities connected to found SH spanners

(d) running time

Fig. 6. Performance of different algorithms in network `Epinions`.



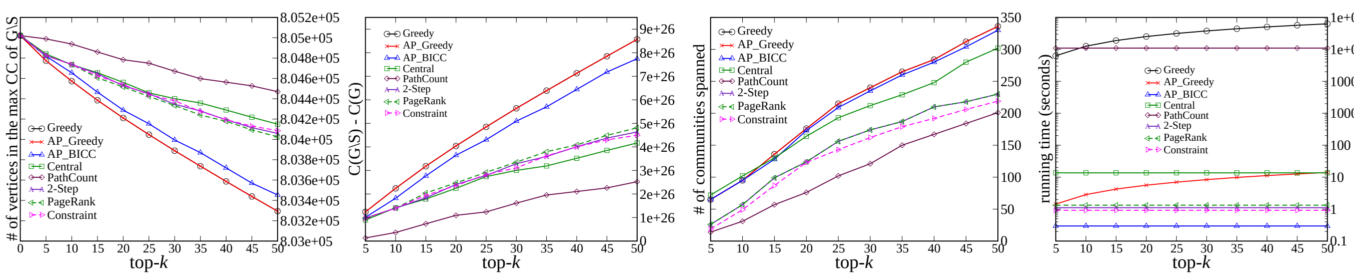
(a) numbers of vertices in the maximum CC of graph $G \setminus S$.

(b) the value of $C(G \setminus S) - C(G)$

(c) numbers of communities connected to found SH spanners

(d) running time

Fig. 7. Performance of different algorithms in network `Twitter`.



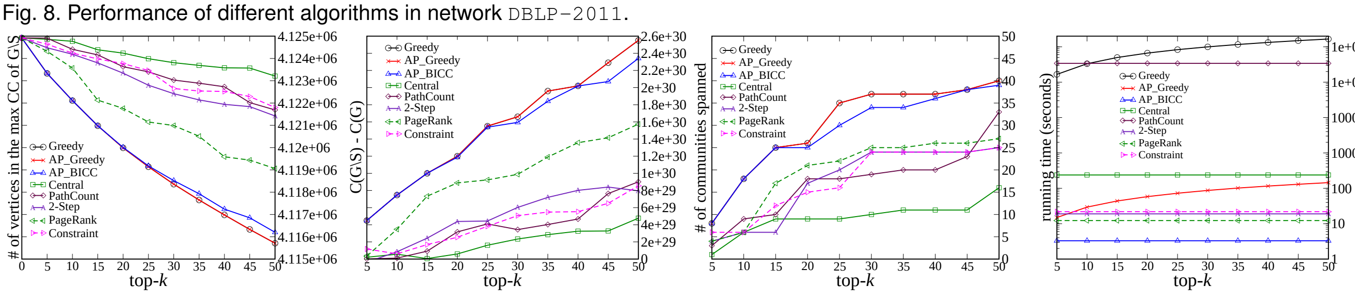
(a) numbers of vertices in the maximum CC of graph $G \setminus S$.

(b) the value of $C(G \setminus S) - C(G)$

(c) numbers of communities connected to found SH spanners

(d) running time

Fig. 8. Performance of different algorithms in network `DBLP-2011`.



(a) numbers of vertices in the maximum CC of graph $G \setminus S$.

(b) the value of $C(G \setminus S) - C(G)$

(c) numbers of communities connected to found SH spanners

(d) running time

Fig. 9. Performance of different algorithms in network `LiveJournal`.

in network `Epinions`, whose size is larger than that of network `GR-QC`. Fig. 6 (a), (b), (c), and (d) demonstrate the performance of the eight mentioned algorithms, in terms of the size of the maximum connected component in graph $G \setminus S$, the value of $C(G \setminus S) - C(G)$, the number of spanned communities, and the running time of each mentioned algorithm, respectively. Similar to the performance of the algorithms in network `GR-QC`, the structural hole spanners found by algorithms `Greedy` and `AP_Greedy` are the best among the eight algorithms. Also, it can be seen that the structural hole spanners identified by algorithm `AP_BICC` are only slightly worse than those by algorithms `Greedy` and `AP_Greedy`.

We finally investigate the performance of the mentioned algorithms in three much larger networks `Twitter`, `DBLP-2011`, and `LiveJournal`, which are plotted in Figures 7, 8, and 9, respectively. Again, we can see that Fig. 7, Fig. 8, and Fig. 9 show that the structural hole spanners delivered by algorithms `Greedy` and `AP_Greedy` are much better than those identified by the other mentioned algorithms. On the other hand, the performance of algorithm `AP_BICC` is similar to that of algorithm `PageRank` in network `Twitter`, but better than the latter in networks `DBLP-2011` and `LiveJournal`.

In summary, it can been seen that the structural hole spanners found by the proposed algorithms `Greedy` and

`AP_Greedy` are much better than those by the other existing algorithms for all mentioned social networks, while the performances of existing algorithms depend on the structure of the social networks. Also, algorithm `AP_Greedy` is highly scalable and its running time in network `LiveJournal` with more than five million vertices and over 27 million edges is only about 145 seconds when $k = 50$ (see Fig. 9 (d)), whereas the running time of algorithm `Greedy` is much longer than the other algorithms.

7 RELATED WORK

Over last few years, the size of real networks has increased enormously, developing efficient algorithms for finding influential individuals in such large-scale networks with unique properties such as structural hole spanners has become a challenging task. Moreover, building accurate models that truly reflect the properties of structural hole spanners is crucial to identify such individuals. Nevertheless, researchers take lots of effort towards this aim.

The notion of structural hole spanners was first introduced by Burt [10] to find the key employees in organizations for integrating operations across functional and business boundaries. This concept later was further refined in [1], [8], [9]. A few studies have exploited the concept of structural holes in order to design strategic games for network formation [14], [20]. Goyal *et al.* [14] presented a network formation model that a vertex u serves as an intermediary between many vertices. However, this strategy leads to the star network and real networks do not follow a star topology. In order to tackle this problem, Kleinberg *et al.* [20] designed a game by building a model of the payoffs that arise from filling structural holes. This payoff is a decreasing function of the number of paths with length two between each pair of neighbors to avoid the star topology. One of the limitations of the model presented by Kleinberg *et al.* [20] is that this model needs careful tuning of parameters such as the link maintenance cost that is not easily achieved in large-scale networks. Another line of research in computer science is to find structural hole spanners in order to incorporate them in contagion, and can be divided in two categories as follows.

Structural-based Models: Goyal *et al.* [14] formulated a structural hole spanner as a vertex that resides on more shortest paths between different pairs of vertices. Since counting the number of shortest paths in large networks is time-consuming, Tang *et al.* [28] proposed to only count the number of shortest paths with length two on which a vertex lies. In this model, any shortest path of length greater than two will be ignored, thus the model suggests candidates that are connected to smaller rather than larger, richer and more influential communities. A fairly common case under this model is its failure of finding good quality structural hole spanners when a vertex is densely connected to two communities. For example, in Fig. 1, vertex v_1 forms more length-2 paths than that of v_2, while v_2 is connected to more communities and is a better structural hole based on Burt's theory [10]. However, this model suggests v_1 as a better structural hole spanner. In order to address this problem, Ugander *et al.* [33] defined the *structural diversity* of an individual as the number of connected components in its contact neighborhood, which is a similar notion as structural hole spanners, and studied the role of structural diversity in contagion of information within real social networks. Huang *et al.* [17] studied the top-k structural diversity search in large networks and developed efficient algorithms for massive dynamic networks. However, only a small number of vertices in each community can be part of contact neighborhood, and they can form multiple connected components. Similarly, Tong *et al.* [32] defined the gateway-ness of a vertex v, proportional to the paths between source vertices S and target vertices T, on which v lies. In addition, each path is given a score, which is inversely proportional to its length.

Community-based Models: Lou *et al.* [21] proposed the very first model to find structural holes in a social network, assuming that communities in the network are given. The objective in their model is to maximize a utility function that measures the degree to which structural hole spanners span communities. One instantiation of their utility function is to find a set of vertices whose removal leads to the maximum decreases on the number of inter-community edges. One major concern about this model is that communities usually are not known, thus the quality of the solution relies on the quality of communities found. Moreover in Fig. 1, the removal of v_1 decreases the number of inter-community edges by 8 and the removal v_2 decreases the inter-community edges by 6. Therefore, this model implies v_1 as the preferred structural hole spanner. However, v_2 should be a better structural hole as it bridges more communities.

8 CONCLUSION

In this paper we studied the top-k structural hole spanner problem in a large social network. We first proposed a novel model to measure the quality of structural hole spanners. We then formulated the problem as an optimization problem and showed its NP-hardness. We thirdly devised two fast yet scalable algorithms for the problem, by developing innovative filtering techniques that can filter out unlikely solutions as early as possible. The developed filtering techniques are capable to provide fast estimations on the upper and lower bounds of the cost of an optimal solution and can explore articulation points in a social network. We finally evaluated the performance of the proposed algorithms and validated the effectiveness of the proposed model, through extensive experiments on real datasets. Experimental results demonstrated that the proposed model can capture the characteristics of structural hole spanners accurately, and the proposed algorithms are very promising, which outperform several other existing heuristics.

REFERENCES

[1] G. Ahuja, "Collaboration networks, structural holes, and innovation: A longitudinal study," *Administ. Sci. Quart.*, vol. 45, no. 3, pp. 425–455, Sep. 2000.

[2] R. Andersen, F. Chuang, and K. Lang, "Local graph partitioning using PageRank vectors," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, 2006, pp. 475–486.

[3] R. Andersen and K. J. Lang, "Communities from seed sets," in *Proc. 15th Int. Conf. World Wide Web (WWW)*, 2006, pp. 223–232.

[4] A. Bar-Noy, S. Khuller, and B. Schieber, "The complexity of finding most vital arcs and nodes," Technical Report CS-TR-35-39, Computer Science Department, University of Maryland, 1995.

[5] M. A. Beauchamp, "An improved index of centrality," *Behavioral Sci.*, vol. 10, no. 2, pp. 161–163, 1965.

[6] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2014, pp. 946–957.

[7] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *Proc. ACM 20th Int. Conf. World Wide Web (WWW)*, 2011, pp. 665–674.

[8] R. S. Burt, "Structural holes and good ideas," *American J. Sociology*, vol. 110, no. 2, pp. 349–399, 2004.

[9] R. S. Burt, "Secondhand brokerage: evidence on the importance of local structure for managers, bankers, and analysts," *Academy Manage. J.*, vol. 50, no. 1, pp. 119–148, Feb. 2007.

[10] R. S. Burt, *Structural holes: The social structure of competition*. Harvard university press, 2009.

[11] S. Chechik, E. Cohen, and H. Kaplan, "Average distance queries through weighted samples in graphs and metric spaces: high scalability with tight statistical guarantees," in *Proc. 18th Int. Workshop Approximation Algorithms Combinatorial Optimization Problems, 19th Int. Workshop Randomization Comput.*, 2015, pp. 659–679.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[13] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proc. Nat. Academy Sci. (PNAS)*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002.

[14] S. Goyal and F. Vega-Redondo, "Structural holes in social networks," *J. Econ. Theory*, vol. 137, no. 1, pp. 460–492, Nov. 2007.

[15] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," *ACM SIGMOD Record*, vol. 42, no. 2, pp. 17–28, Jul. 2013.

[16] L. He, C. T. Lu, J. Ma, J. Cao, L. Shen, and P. S. Yu, "Joint community and structural hole spanner detection via harmonic modularity," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 875–884.

[17] X. Huang, H. Cheng, R.-H. Li, L. Qin, and J. X. Yu, "Top-k structural diversity search in large networks," in *Proc. VLDB Endowment*, vol. 6, no. 13, pp. 1618–1629, 2013.

[18] U. Kang and C. Faloutsos, "Beyond'caveman communities': Hubs and spokes for graph compression and mining," in *Proc. IEEE 11th Int. Conf. Data Mining*, 2011, pp. 300–309.

[19] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 137–146.

[20] J. Kleinberg, S. Suri, É. Tardos, and T. Wexler, "Strategic network formation with structural holes," in *Proc. 9th ACM Conf. Electron. Commerce*, 2008, pp. 284–293.

[21] T. Lou and J. Tang, "Mining structural hole spanners through information diffusion in social networks," in *Proc. ACM 22nd Int. Conf. World Wide Web (WWW)*, 2013, pp. 825–836.

[22] M. V. Marathe and A. K. S. Vullikanti, "Computational epidemiology," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 1969–1969.

[23] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: optimal sampling algorithms for viral marketing in billion-scale networks," in *Proc. ACM Int. Conf. Manage. Data (SIGMOD)*, 2016, pp. 695–710.

[24] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." 1999.

[25] J. Plesník, "On the sum of all distances in a graph or digraph," *J. Graph Theory*, vol. 8, no. 1, pp. 1–21, Mar. 1984.

[26] M. Rezvani, W. Liang, W. Xu, and C. Liu, "Identifying top-k structural hole spanners in large-scale social networks," in *Proc. 24th ACM Int. Conf. Inform. Knowl. Manage.*, 2015, pp. 263–272.

[27] E. Rinia, T. V. Leeuwen, E. Bruins, H. V. Vuren, and A. V. Raan, "Citation delay in interdisciplinary knowledge exchange," *Scientometrics*, vol. 51, no. 1, pp. 293–309, 2001.

[28] J. Tang, T. Lou, and J. Kleinberg, "Inferring social ties across heterogeneous networks," in *Proc. 5th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2012, pp. 743–752.

[29] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proc. 2015 ACM Int. Conf. Manage. Data (SIGMOD)*, 2015, pp. 1539–1554.

[30] J. Tang, S. Wu, and J. Sun, "Confluence: Conformity influence in large social networks," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2013, pp. 347–355.

[31] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proc. 2014 ACM Int. Conf. Manage. Data (SIGMOD)*, 2014, pp. 75–86.

[32] H. Tong, S. Papadimitriou, C. Faloutsos, S. Y. Philip, and T. Eliassi-Rad, "Gateway finder in large graphs: problem definitions and fast solutions," *Inform. Retrieval*, vol. 15, no. 3-4, pp. 391–411, Feb. 2012.

[33] J. Ugander, L. Backstrom, C. Marlow, and J. Kleinberg, "Structural diversity in social contagion," *Proc. Nat. Academy Sci. (PNAS)*, vol. 109, no. 16, pp. 5962–5966, Apr. 2012.

[34] L. Wang, T. Lou, J. Tang, and J. E. Hopcroft, "Detecting community kernels in large social networks," in *IEEE 11th Int. Conf. Data Mining (ICDM)*, 2011, pp. 784–793.

[35] W. Xu, W. Liang, X. Lin, and J. X. Yu, "Finding top-k influential users in social networks under the structural diversity model," *Inform. Sci.*, vol. 355–356, pp. 110–126, Aug. 2016.

[36] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, pp. 181-213, 2015.

Wenzheng Xu (M'15) received the BSc, ME, and PhD degrees in computer science from Sun Yat-Sen University, Guangzhou, P.R. China, in 2008, 2010, and 2015, respectively. He currently is a Special Associate Professor at Sichuan University and was a visitor at the Australian National University. His research interests include wireless ad hoc and sensor networks, online social networks, mobile computing, approximation algorithms, combinatorial optimization, and graph theory. He is member of the IEEE.

Mojtaba Rezvani received his MSc degree from Amirkabir University of Technology in Iran in 2012 and received his BSc from University of Qom in 2010, both in Computer Science. He is currently pursuing his PhD study in the Research School of Computer Science at the Australian National University. His research interests include graph databases, community detection, social networks, database, graph theory and algorithm design.

Weifa Liang (M'99–SM'01) received the PhD degree from the Australian National University in 1998, the ME degree from the University of Science and Technology of China in 1989, and the BSc degree from Wuhan University, China in 1984, all in computer science. He is currently a professor in the Research School of Computer Science at the Australian National University. His research interests include design and analysis of energy-efficient routing protocols for wireless ad hoc and sensor networks, cloud computing, software-defined networking, online social networks, design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He is a senior member of the IEEE.

Jeffrey Xu Yu received the BE, ME, and PhD degrees in computer science, from the University of Tsukuba, Tsukuba, Japan, in 1985, 1987, and 1990, respectively. He is currently a professor in the Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong. His current research interests include graph mining, graph database, social networks, keyword search, and query processing and optimization. He is a senior member of the IEEE, a member of the IEEE Computer Society, and a member of the ACM.

Chengfei Liu received the BS, MS and PhD degrees in Computer Science from Nanjing University, China in 1983, 1985 and 1988, respectively. Currently, he is a Professor in Swinburne University of Technology, Australia. His research interests include keywords search on structured data, query processing and refinement for advanced database applications, query processing on uncertain data and big data, and data-centric workflows. He is a member of IEEE and ACM.