Information Sciences 505 (2019) 100–126

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Identifying structural hole spanners to maximally block information propagation[☆]

Wenzheng Xu[a], Tong Li[a], Weifa Liang[b], Jeffrey Xu Yu[c], Ning Yang[a], Shaobing Gao[a,*]

[a] College of Computer Science, Sichuan University, Chengdu 610065, PR China
[b] The Australian National University, Canberra ACT0200, Australia
[c] The Chinese University of Hong Kong, Hong Kong, PR China

A R T I C L E I N F O

Article history:
Received 7 November 2018
Revised 17 July 2019
Accepted 19 July 2019
Available online 19 July 2019

Keywords:
Social networks
Top-k structural hole spanners
Block information propagations
Approximation algorithm

A B S T R A C T

An individual can obtain high profits by playing a bridge role among different communities in a social network, thus acquiring more potential resources from the communities or having control over the information transmitted within the network. Such an individual usually is referred to as a structural hole spanner in the network. Existing studies on the identification of important structural hole spanners focused on only whether a person bridges multiple communities without emphasizing the tie strength of that person connecting to his bridged communities. However, a recent study showed that such tie strength heavily affects the profit the person obtains by playing the bridge role. In this study, we aim to identify the most important hole spanners in a large-scale social network who have strong ties with their bridged communities. Accordingly, we first formulate the top-k structural hole spanner problem to identify k nodes in the network such that their removals will block the maximum numbers of information propagations within the network. Due to the NP-hardness of the problem, we then propose a novel $(1-\epsilon)$-approximation algorithm, where ϵ is a given constant with $0<\epsilon<1$. Although the approximate solution provides a guaranteed performance, it takes some time to deliver the solution, which may not be acceptable in practice. Therefore, we devise a fast, yet scalable, heuristic algorithm for the problem. Finally, we evaluate the performance of the proposed algorithms through extensive experiments, using real-world datasets. The experimental results show that the proposed algorithms are very promising. Especially, the found hole spanners block more than 24% of the information propagations compared to existing algorithms.

1. Introduction

A variety of large-scale networks have sprung up in the past decades, e.g., online social networks, collaboration networks, World Wide Web (WWW), and biological networks [10,21,34]. These networks have a wide range of applications, such as social network analysis, web link analysis, computational biology, and epidemic disease control [10,13,21,24,28,31,36]. Most

☆ This document is a collaborative effort.
* Corresponding author.

E-mail addresses: wenzheng.xu3@gmail.com (W. Xu), tong.li.scu@foxmail.com (T. Li), wliang@cs.anu.edu.au (W. Liang), yu@se.cuhk.edu.hk (J.X. Yu), yangning@scu.edu.cn (N. Yang), gaoshaobing@scu.edu.cn (S. Gao).

https://doi.org/10.1016/j.ins.2019.07.072
0020-0255/© 2019 Elsevier Inc. All rights reserved.

such networks exhibit the so-called community structure [12,22,35,38,39]. That is, nodes within the same community are densely connected. A node is more likely to connect to other nodes within its community and less likely to contact the nodes outside its community. As a result, nodes in the same community usually share highly similar attributes. For example, in a social network, users in the same community have similar backgrounds, opinions, or behavior.

A famous sociologist Burt showed that a person who bridges multiple communities can acquire many more potential resources. Hence, the person has greater competition advantages than those of another person who only belongs to his community [3,5,6]. Because a person usually communicates more with people in his community than with the people outside, the information from the people in the same community tends to be homogeneous and redundant, while the information obtained from the people in other communities is very likely to be nonredundant and diverse. Therefore, a person who connects with multiple communities can obtain *high profits* by acquiring more nonredundant pieces of information, receiving important information from other communities earlier, or arbitrating passed information [3,5]. A person bridging multiple communities is usually referred to as a *structural hole spanner* [23].

The detection of important structural hole spanners in a large-scale social network has many important applications. For example, it is well-known that fake news and rumors frequently spread on social networks. Even worse, such misinformation propagates six times faster than the truth on Twitter [32]. We can identify only a few important structural holes to prevent the wide spread of rumors efficiently, by filtering the misinformation passing through them, rather than by quarantining a large number of persons [2,24,31,36]. In addition, a structural hole researcher in an academic collaboration network is interested in several research fields. The person could apply ideas and techniques from one field to the problems faced by the other fields or innovate by synthesizing ideas from different communities, thus creating interdisciplinary studies [5,34].

Several studies on the detection of important structural hole spanners in social networks have been conducted [16,23,27,34]. Most of the studies identified such spanners by exploiting the topological characteristics of structural holes. For example, Lou and Tang [23] argued that the removal of important structural hole spanners can significantly decrease the minimum cut of the communities, where such cut is the minimum number of edges such that their removals separate each community from its linked communities. Rezvani et al. [27] and Xu et al. [34] observed that the removal of a node bridging multiple communities will significantly increase the communication cost, where such cost is referred to the sum of all-pairs shortest distances. He et al. [16] devised a harmonic modularity method to jointly detect communities and structural hole spanners.

1.1. Motivations

Despite of the aforementioned studies on the identification of important structural hole spanners, the pioneering sociologist Burt [7] recently showed that although structural hole spanners have higher profits on average than those of other peers who communicate only with the people in their own communities, the profits obtained by different hole spanners vary widely, i.e., some hole spanners have high profits, while the others have much lower profits.

To bring a high profit to a person, Burt [7] found that the person should not only bridge multiple communities to obtain nonredundant information but also need to build *strong ties* with his bridged communities. Here, a strong tie between the person and one of his bridged communities means that the person has many friends in the community, the person has a close relationship with each of the friends (e.g., frequent contacts), or one of his friends is an opinion leader in the community. Here, ideas and innovations usually first flow to opinion leaders, and then from the opinion leaders to a wider, yet less active, population [17,23]. By building strong ties with his bridged communities, the person can obtain more reliable and valuable information from the communities, and his information is trusted and his developed ideas are easily accepted by the community members [4,7]; hence, the person obtains a high profit by acting the bridge role.

We use an example here to illustrate the importance of the tie strength between a structural hole spanner and his bridged communities. Consider preventing the wide spread of misinformation, e.g., rumors, in a social network (see Fig. 1), by filtering the misinformation passing through one user, rather than through every user in the social network. There are 16 users in the network, and these users form four communities $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$. User *J* bridges three communities $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, while user *F* bridges only two communities $\mathcal{C}_3$ and $\mathcal{C}_4$, where a thick and a thin line between two users in Fig. 1 indicate their close and weak relationships, respectively. It can be easily seen that both users *J* and *F* bridge multiple communities. Thus, they are structural hole spanners in the network. A critical question is which of the two users is the more important hole spanner?

Existing studies will identify user *J* as the more important structural hole spanner, as user *J* brides more communities than those by user *F*. However, it can be seen that the relationship between user *J* and each of his bridged communities $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$ is weak, and thus only a limited amount of information is transmitted among the three communities through user *J*. Therefore, the monitoring of user *J* will filter a limited amount of misinformation.

In contrast, although user *F* bridges only two communities $\mathcal{C}_3$ and $\mathcal{C}_4$, which is less than that by user *J*, he builds close relationships with both communities $\mathcal{C}_3$ and $\mathcal{C}_4$. Then, a large amount of information propagates between $\mathcal{C}_3$ and $\mathcal{C}_4$ via user *F*. Therefore, compared with the quarantining of user *J*, the quarantining of user *F* will block more misinformation propagations.

From the example in Fig. 1, it can be seen that, although the hole spanners identified by existing studies bridge many communities [16,23,27,34], they might have had only weak ties with their bridged communities (e.g., user *J* in Fig. 1). There-

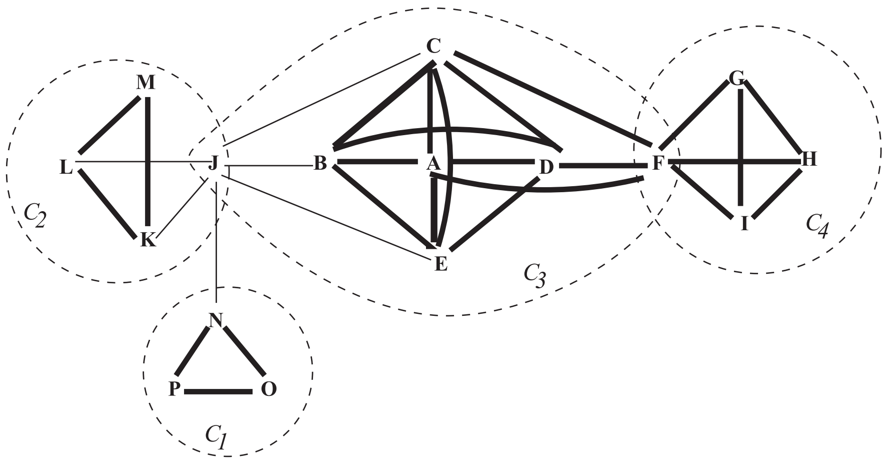


Fig. 1. Illustration of the importance of the strength of the ties that connect a structural hole spanner to his bridged communities, where there are four communities $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ in a social network, user J bridges three communities $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, while user F bridges only two communities $\mathcal{C}_3$ and $\mathcal{C}_4$, a thick and a thin line between two users indicate their close and weak relationships, respectively.

fore, the hole spanners found by them may not play important roles at all. For example, the found structural hole users in social networks may be ineffective in rumor control [2,24,36].

Motivated by the aforementioned concerns, in this paper, we aim to find important structural hole spanners that not only bridge multiple communities but also have strong ties with their bridged communities. We observe that, for a structural hole spanner having strong ties with his bridged communities, the removal of the spanner will heavily block information propagations within the network, while the removal of another hole spanner with weak ties to his bridged communities does not have such an impact. We thus define the top-k structural hole spanners in a network as the k nodes whose removals will block the maximum number of information propagations within the network. Therefore, user F in Fig. 1 will be considered as a much more important hole spanner than user J.

1.2. Difference from the influence maximization problem

Intuitively, the top-k structural hole spanner problem considered in this paper has some similarities to the influence maximization problem [18,20,30,33,37]; they are, however, essentially different, which is shown as follows.

Given a social network, the influence maximization problem aims to find k users such that the expected number of users receiving a piece of information originating from the k users is maximized [18]. The main difference is that an important structural hole spanner must bridge multiple communities, while an influential person may communicate only with people in his community. For example, user A in Fig. 1 is the most influential user, as he lies at the center of the network. However, the removal of user A only slightly affects the information propagations of others, as this user is in a single community. In contrast, although user F is less influential than user A, its removal will block many information propagations because he bridges two communities. Our later empirical results also validate the fact that the number of blocked information propagations by the most influential users is much lower than that by the most important hole spanners using real-world datasets.

1.3. Novelty and contributions

To the best of our knowledge, this is the first study to identify important structural hole spanners that not only bridge multiple communities, but also have strong ties with their bridged communities. We also develop a performance-guaranteed approximation and a fast heuristic algorithms for the problem.

The main contributions of this paper are summarized as follows:

- First, we formulate a novel top-k structural hole spanner problem, which aims to find k users in a social network such that their removals will block the maximum number of information propagations within the network.
- We then propose a randomized algorithm for the problem, which delivers a $(1-\epsilon)$-approximate solution with a high probability $1-\frac{1}{n^c}$, whose expected running time is $O(\frac{\epsilon^{-2}(k+c)k^2n^2m\log n}{OPT})$, where ϵ and c are two given constants with $0<\epsilon<1$ and $c\geq 1$, respectively; n and m are the numbers of nodes and edges in the network, respectively; and OPT is the value of the optimal solution to the problem.
- Although the approximate solution provides a guaranteed performance, it takes some time to deliver the solution. Therefore, we devise a fast, yet scalable, heuristic algorithm for the problem by exploring the combinatorial properties of the problem.

- Finally, we evaluate the performance of the proposed algorithms through extensive experiments, using real-world datasets. The experimental results show that the two proposed algorithms are very promising. Especially, their found hole spanners block more than 24% of the information propagations compared to that by existing algorithms, and the hole spanners identified by the heuristic algorithm block only slightly fewer information propagations than those by the approximation algorithm; however, the running time of the former is approximately 13 to 300 times shorter than that of the latter.

The rest of this paper is organized as follows. Section 2 introduces the system model and defines the problem. Section 3 proposes a randomized approximation algorithm for the problem, and Section 4 presents the analysis of the performance of this algorithm. Section 5 devises a fast heuristic algorithm for the problem. Section 6 presents the evaluation of the algorithm performance through extensive experiments. Section 7 reviews related studies, and finally, Section 8 concludes this paper.

2. Preliminaries

In this section, we first introduce the network model and the information propagation model. We then define the problem and show the important properties of the problem.

2.1. Network model

A social network can be represented as a directed weighted graph $G=(V,E)$, where V is the set of nodes representing the users in the network. For two users v_i and v_j in V, there is a directed edge $(v_i, v_j)\in E$ from v_i to v_j if user v_j follows user v_i. Note that the proposed algorithms in this paper are also applicable to undirected social networks, by replacing each undirected edge $\{v_i, v_j\}$ with two directed edges: (v_i, v_j) and (v_j, v_i). Denote by n and m the numbers of nodes and edges in G, respectively, i.e., $n=|V|$ and $m=|E|$.

Each user $v_i\in V$ can share his/her information with others, such as ideas, opinions, or photos [18,33], on the social network. Assume that user v_i shares his/her information at an average rate of f_i with $f_i>0$. Without loss of generality, assume that $f_i \leq 1$ for each node $v_i\in V$. Otherwise ($f_i>1$ for some nodes), we can scale down the value of f_i of each node v_i by multiplying a factor of $\frac{1}{f_{\max}}$, where $f_{\max}=\max_{v_j\in V}\{f_j\}$.

The weight $w_{i,j}$ of each edge (v_i, v_j) in E represents *the activation probability* from node v_i to v_j, which indicates the extent to which user v_i influences user v_j. The value of $w_{i,j}$ can be obtained from the information diffusion history as follows. Assume that user v_i sent $m_{i,j}$ pieces of information to user v_j, while user v_j then forwarded $m'_{i,j}$ of the $m_{i,j}$ pieces of information to his friends. Then, $w_{i,j}=\frac{m'_{i,j}}{m_{i,j}}$, where $0\leq w_{i,j}\leq 1$.

For any node $v\in V$, denote by $N_{out}(v)$ and $N_{in}(v)$ its outgoing neighbor and incoming neighbor sets in G, respectively, i.e., $N_{out}(v)=\{u \mid u\in V, (v,u)\in E\}$ and $N_{in}(v)=\{u \mid u\in V, (u,v)\in E\}$. Let $G[V\backslash\{v\}]$ be the induced subgraph of G by the nodes in $V\backslash\{v\}$. We abbreviate $G[V\backslash\{v\}]$ as $G\backslash\{v\}$.

Given a rooted node s in graph $G=(V,E)$, assume that every other node $v\in V\backslash\{s\}$ is reachable from s. For any two nodes u and v in V with $u\neq v$, node u *dominates* v in G if every path from s to v contains u. Node u is the *immediate dominator* of node v, which is denoted by $u=idom(v)$, if u dominates v and every other dominator of v dominates u. Each node $v\in V$ except s may have multiple dominators, but has only one immediate dominator [11]. The edges $\{(idom(v), v) \mid v\in V\backslash\{s\}\}$ form a directed tree T_{dom} rooted at s, which is referred to as a *dominator tree* of G.

2.2. The independent cascade model [18]

We adopt a well-known information diffusion model, namely, the independent cascade model [18], which is defined as follows.

Given a directed graph $G=(V,E)$, an activation probability function w: $E\mapsto[0, 1]$, and a seed node $s\in V$, the information diffusion process under the *independent cascade model* starting from node s proceeds as follows.

Denote by A_t the set of activated nodes at time t, where $t=0,1,2,\ldots,$ and $A_0=\{s\}$. At time $t+1$, initially, let $A_{t+1}=A_t$, which means that once a node is activated, it will stay active thereafter. For each newly activated node $v_i\in A_t\setminus A_{t-1}$, it has a single chance to activate an inactive out-neighbor $v_j\in N_{out}(v_i)\cap(V\backslash A_t)$ with probability w_{ij}. Node v_j is added to A_{t+1} if it is activated by node v_i successfully. The information propagation continues until $A_{t+1}=A_t$ for some time t.

For any two given nodes v_i and v_j in G, denote by $p_{i,j}$ *the information propagation probability* from user v_i to user v_j, which is the probability that v_j receives a piece of information originating from v_i with $i\neq j$. For example, the information propagation probability $p_{1,2}$ from users v_1 to v_2 in Fig. 2 is $p_{1,2}=1-(1-w_{1,3}\cdot w_{3,2})(1-w_{1,4}\cdot w_{4,2})=1-(1-0.2\cdot 0.5)(1-0.1\cdot 0.4)=0.136$.

Although it is not difficult to calculate the information propagation probability between any pair of nodes in a small social network, such calculation becomes very challenging for a large-scale network, because there may be an exponential number of different paths between a pair of nodes. In fact, the problem of calculating the information propagation probability between each pair of nodes is NP-hard [9]. Fortunately, Kempe et al. demonstrated that the independent cascade

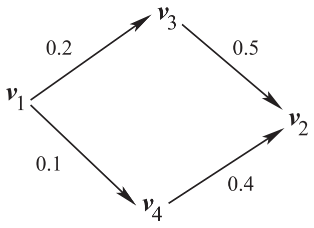


Fig. 2. Illustration of the calculation of the information propagation probability $p_{1,2}$, where $p_{1,2} = 1 - (1 - w_{1,3} \cdot w_{3,2})(1 - w_{1,4} \cdot w_{4,2}) = 1 - (1 - 0.2 \cdot 0.5)(1 - 0.1 \cdot 0.4) = 0.136$.

model is equivalent to another important model, namely, the live-edge graph model [18]. Through Monte Carlo simulations, the information propagation probability between a pair of nodes can be estimated with a relatively small error under the live-edge graph model, which is defined as follows.

Given a graph $G = (V, E)$, an activation probability function w: $E \mapsto [0, 1]$, and a seed node $s \in V$, an unweighted graph $G' = (V, E')$ is derived from G as follows. For each edge $(v_i, v_j) \in E$, it is contained in edge set E' with probability $w_{i,j}$. For any two nodes $u, v \in V$, denote by $d_{G'}(u, v)$ the distance of the shortest path in G' from node u to node v, assuming that the length of each edge in G' is one. Moreover, denote by $R_{G'}(s, t)$ the set of nodes within t hops from node s in G', i.e., $R_{G'}(s, t) = \{v \mid v \in V, d_{G'}(s, v) \le t\}$, where t is a nonnegative integer. Under *the live-edge graph model*, the set of active nodes at time t is $R_{G'}(s, t)$, where $t = 0, 1, \ldots$.

The equivalence between the independent cascade model and the live-edge graph model is shown by the following lemma.

Lemma 1 [9]**.** *Given a graph $G = (V, E)$, w: $E \mapsto [0, 1]$, a seed node s, let $G' = (V, E')$ be a graph generated from G under the live-edge graph model. Assume that $A_0 = R_{G'}(s, 0)$, $A_1 = R_{G'}(s, 1)$, ..., $A_t = R_{G'}(s, t)$ for each time t. Then, for each inactive node $u \in V \backslash A_t$, the probability that node u is activated at time $t + 1$ under the independent cascade model is equal to the probability that it is influenced at time $t + 1$ under the live-edge graph model, i.e., $Pr[u \in A_{t+1}] = Pr[u \in R_{G'}(s, t + 1)]$.*

2.3. Problem definition

Recall that, for any two users v_i and v_j in G, $p_{i,j}$ is the probability that v_j receives a piece of information originating from node v_i. Then, the expected pieces of information per unit time propagated from user v_i to user v_j are $f_i \cdot p_{i,j}$, where user v_i shares his/her information on network G at a rate f_i.

Given a potential structural hole spanner u with $u \ne v_i$ and $u \ne v_j$, the expected pieces of information propagating from v_i to v_j after the removal of u are $f_i \cdot p'_{i,j}$, where $p'_{i,j}$ is the probability that a piece of information successfully propagates from user v_i to user v_j in $G \backslash \{u\}$. For example, Fig. 2 shows that the information propagation probability $p'_{1,2}$ after the removal of node v_3 is $p'_{1,2} = w_{1,4} \cdot w_{4,2} = 0.1 \cdot 0.4 = 0.04$, which is smaller than the information propagation probability $p_{1,2} = 0.136$ before the removal of v_3, i.e., $p'_{1,2} = 0.04 < 0.136 = p_{1,2}$.

The expected number $H(u)$ of blocked information propagations by the removal of user u is thus

$$H(u) = \sum_{v_i \in V \backslash \{u\}} \sum_{v_j \in V \backslash \{v_i, u\}} (f_i p_{i,j} - f_i p'_{i,j}) = \sum_{v_i \in V \backslash \{u\}} f_i \sum_{v_j \in V \backslash \{v_i, u\}} (p_{i,j} - p'_{i,j}). \tag{1}$$

Given a social network $G = (V, E)$, $f : V \mapsto \mathcal{R}^+$, w: $E \mapsto [0, 1]$, and a positive integer $k \le |V|$, *the top-k structural hole spanner problem* in G is to identify a set $S (\subseteq V)$ of k users such that the sum of the numbers of blocked information propagations by the users in S is maximized, i.e.,

$$S = \arg \max_{V' \subseteq V, |V'| = k} \{H(V')\}, \tag{2}$$

where $H(V') = \sum_{u \in V'} H(u)$.

The top-k structural hole spanner problem is NP-hard, see Appendix A.

2.4. Important properties of the problem

The problem definition for the top-k structural hole spanners exhibits the following four important properties.

(1) **Correctness**. A user u bridging multiple communities is considered as an important structural hole spanner. This is because the removal of the user will significantly block information diffusions among the communities bridged by the user. In contrast, the removal of another user v who communicates only with the people in his own community barely affects the communication of other people.

(2) **Strong ties**. A user u who has strong ties with the people in his bridged communities is considered as a more important hole spanner than another user v who has only weak ties with his connected communities, as user u can obtain more information from his bridged communities and the information is more likely to propagate to the members in his

connected communities, while user v might obtain little information from his bridged communities and his information is less likely to diffuse to the members in his connected communities.

(3) **Diversity**. A user u who connects to many and large communities, instead of another user v who connects only to a few or small communities, is identified as a better hole spanner, since the removal of u will block more information propagations than that by v.

(4) **Uniqueness**. Given some communities in a social network, there may be multiple users bridging the communities, and these users are considered as alternative structural hole spanners with each other. It can be seen that the fewer these alternative hole spanners are, the more information propagations will be blocked by the removal of one of them. A user u who has a few alternative hole spanners bridging his connected communities is regarded as a better hole spanner than another user v who has many alternative ones.

3. Approximation algorithm

In this section, we propose a randomized algorithm for the top-k structural hole spanner problem that delivers a $(1-\epsilon)$-approximate solution with a high probability $1-n^{-c}$, where ϵ and c are two given constants with $0<\epsilon<1$ and $c\geq 1$, respectively.

3.1. Basic idea of the algorithm

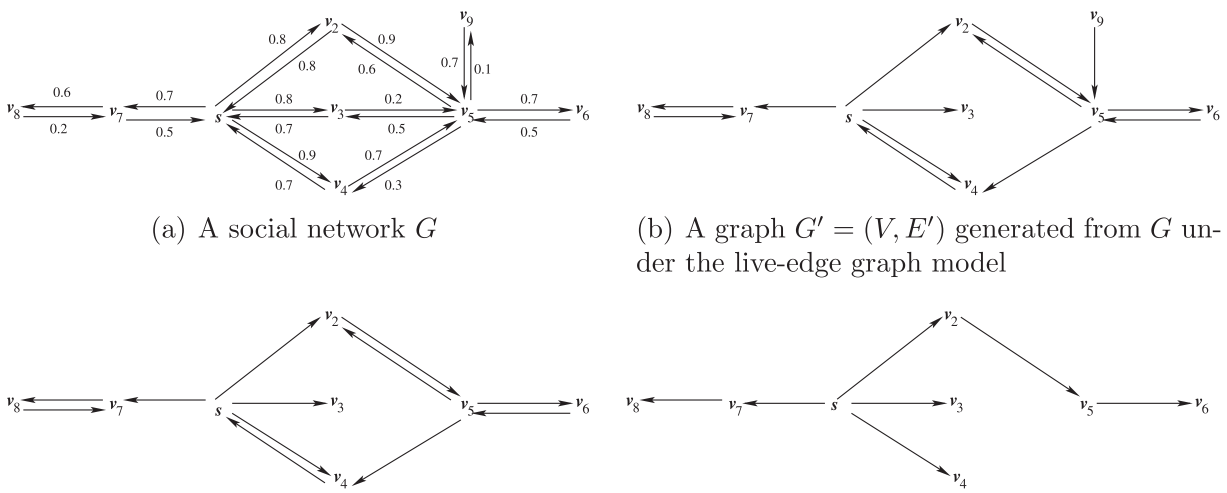
The basic idea of the approximation algorithm is to first simulate a sufficiently large number L of information propagations under the live-edge graph model, and then calculate the average number of blocked information propagations by the removal of each user in the L simulations. Then, identify the k users who block the maximum average numbers of information propagations as the top-k structural hole spanners. In the development of the algorithm, the following three issues must be addressed.

(i) Given the L times of simulations, how do we calculate the average number of blocked information propagations by the removal of each user?
(ii) What is the minimum number $L_{\min}$ of simulations needed for finding the top-k structural hole spanners?
(iii) How do we identify the top-k users who block the maximum numbers of information propagations?

In the following, we address the aforementioned three issues one by one.

3.2. Calculation of the average number of blocked information propagations by each user under a given number of simulations

Given L times of simulations, in the lth simulation with $1\leq l\leq L$, the algorithm first chooses a node s from the $n(=|V|)$ nodes randomly. It then randomly constructs an auxiliary directed graph $G'=(V,E')$ from the original graph $G=(V,E)$ with w: $E\mapsto[0,1]$ under the live-edge graph model (see Section 2.2). For example, Fig. 3(a) shows a social network G consisting of nine nodes $s, v_2, v_3, \ldots, v_8$, and Fig. 3(b) illustrates a graph $G'=(V,E')$ generated under the live-edge graph model.



(a) A social network G

(b) A graph $G'=(V,E')$ generated from G under the live-edge graph model

(c) The induced subgraph $G''=(V'',E'')$ of G' by $R_{G'}(s)$

(d) The s-rooted dominator tree T_{dom} of graph G''

Fig. 3. Illustration of the information propagation sampling.

Denote by $R_{G'}(s)$ the set of reachable nodes from node s in G', i.e., $R_{G'}(s) = \{u \mid u \in V,$ and there is a directed path in G' from s to $u\}$. Following Lemma 1, the probability that a piece of information originating from s propagates to a node $v \in V$ under the independent cascade model is equal to the probability that node v is contained in set $R_{G'}(s)$ under the live-edge graph model. Later we will show that the probability that the removal of a node $u \in V$ from G blocks the information propagation from node s to a node v under the independent cascade model, is equal to the probability that node v is reachable from node s in G' but becomes unreachable in graph $G' \backslash \{u\}$.

Consider a piece of information from s to be broadcast; for each candidate structural hole node $u \in V \backslash \{s\}$, we calculate the number of blocked nodes by removing node u in G' as follows. We first obtain the induced subgraph $G'' = (V'', E'')$ of $G' = (V, E')$ by set $R_{G'}(s)$, i.e., $V'' = R_{G'}(s)$, and $E'' = \{(u, v) \mid (u, v) \in E', u, v \in R_{G'}(s)\}$, (see Fig. 3(c)), where node v_9 is not contained in G'' since it is unreachable from s in G'. We consider only the set of reachable nodes $R_{G'}(s)$ from s, as any node $v' \in V \setminus R_{G'}(s)$ is unreachable from s, and thus, its removal does not block any information propagation from s.

We then find the immediate dominator $idom(v)$ of each node $v \in V'' \backslash \{s\}$, by invoking an algorithm in [11]. For example, in Fig. 3(c), the immediate dominators of nodes $v_2, v_3, \ldots, v_8$ are $s = idom(v_2), s = idom(v_3), s = idom(v_4), v_2 = idom(v_5), v_5 = idom(v_6), s = idom(v_7)$, and $v_7 = idom(v_8)$, respectively. We can construct an s-rooted dominator tree T_{dom} of G'' with the set of edges $\{(idom(v), v) \mid v \in V'' \backslash \{s\}\}$ (see Fig. 3(d)).

It can be seen that the removal of a node u with $u \neq s$ from G'' will block the information propagation from s to the proper descendants of node u in the tree T_{dom}. For instance, in Fig. 3(d), the removal of node v_2 blocks the information diffusion from s to both nodes v_5 and v_6.

Denote by X_l^u the number of blocked information propagations by node u in the lth simulation with $1 \leq l \leq L$, and denote by n_u the number of proper descendants of node u in the tree T_{dom} rooted at s. Then, $X_l^u = f_s \cdot n_u$, where node s shares his information at a rate of f_s. For instance, in Fig. 3(d), the numbers of blocked information propagations by the seven nodes v_2, v_3, v_4, v_5, v_6, v_7, and v_8 are $2f_s$, 0, 0, f_s, 0, f_s, and 0, respectively.

The average number of blocked information propagations $\overline{H_L(u)}$ within G by node u in the L simulations then is

$$\overline{H_L(u)} = \frac{\sum_{l=1}^{L} X_l^u}{L} n, \tag{3}$$

since $\frac{\sum_{l=1}^{L} X_l^u}{L}$ is the average number of information propagations blocked by u from a randomly chosen node s, and there are n nodes in the network.

The algorithm for finding the top-k structural hole spanners under L simulations is presented in Algorithm 1.

Algorithm 1 Find the top-k structural hole spanners under L simulations.

```
Input: A social network $G = (V, E)$, $f : V \mapsto \mathcal{R}^+$, $w : E \mapsto [0, 1]$, a positive integer $k$, and the number of simulations $L$.
Output: A set $S$ of $k$ nodes that block the maximum number of information propagations within $L$ simulations
 1: For each $u \in V$, let $h_u \leftarrow 0$ ; /* blocked information propagations by each node */
 2: for $l \leftarrow 1$ to $L$ do
 3:     Choose a node $s$ from the $n$ nodes in $G$ randomly;
 4:     Generate a graph $G' = (V, E')$ from $G$ under the live-edge graph model;
 5:     Obtain the induced subgraph $G'' = (V'', E'')$ by the set $R_{G'}(s)$ of reachable nodes from $s$;
 6:     Find the immediate dominator $idom(v)$ of each node $v \in V'' \setminus \{s\}$ in $G''$;
 7:     Construct an $s$-rooted dominator tree $T_{dom}$ of $G''$ with the set of edges $\{(idom(v), v) | v \in V'' \setminus \{s\}\}$, by invoking an algorithm in [11];
 8:     Calculate the number of proper descendants $n_u$ of each node $u \in V''$ in the tree $T_{dom}$ by performing a depth-first search starting from root $s$;
 9:     for each node $u \in T_{dom}$ with $u \neq s$ do
10:         $h_u \leftarrow h_u + f_s \cdot n_u$;/* the number of blocked information propagations by node $u$ */
11:     end for
12: end for
13: Let $\overline{H_L(u)} \leftarrow \frac{h_u}{L} n$ for each node $u \in V$;
14: Sort the nodes in $V$ in nonincreasing order by their values $\overline{H_L(u)}$, assume that $\overline{H_L(u_1)} \geq \overline{H_L(u_2)} \geq \cdots \geq \overline{H_L(u_n)}$, and let $S = \{u_1, u_2, \ldots, u_k\}$.
```

3.3. Estimation of the minimum number of simulations

We now show that the minimum number of simulations $L_{\min}$ needed for finding a $(1 - \epsilon)$-approximate solution with high probability is highly correlated with the optimal value OPT of the problem. The smaller OPT is, the more simulations $L_{\min}$ is needed. That is, $L_{\min} = \frac{\gamma}{OPT}$, where γ is a parameter that does not vary with the change of OPT (see Lemma 7 in Section 4 for more details).

Since the calculation of OPT is NP-hard, a simple method for finding the minimum number of simulations $L_{\min}$ is to conservatively adopt a small value Δ (e.g., $\Delta = 1$), presume that Δ is a lower bound on OPT, and then set $L = \frac{\gamma}{\Delta}$. Therefore,

$L = \frac{\gamma}{\Delta} \geq L_{\min} = \frac{\gamma}{OPT}$. However, the number of simulations L obtained by this method is very large and a considerably long time will be spent on finding the approximate solution. In the following, we show how to obtain an approximate value OPT' of OPT so that $\frac{OPT}{8} \leq OPT' \leq OPT$ with high probability. Consequently, $L_{\min} = \frac{\gamma}{OPT'} (\geq \frac{\gamma}{OPT})$ is much smaller than $\frac{\gamma}{\Delta}$, especially when OPT is very large, i.e., $\frac{\gamma}{OPT'} \ll \frac{\gamma}{\Delta}$ if $OPT \gg \Delta$. For example, the OPT in our later experiments is usually as large as the number of nodes in the social network being dealt with.

Having an OPT', the approximation algorithm then determines the minimum number of simulations $L_{\min}$ by setting $L_{\min} = \frac{\epsilon^{-2}((k+c)\log n + \log 4)(8k+2\epsilon)kn^2}{OPT'}$.

The rest is to estimate the value of OPT. Let $OPT_{ub} = k(n-1)\sum_{i=1}^{n} f_i$, which is an upper bound on OPT, since the number of blocked information propagations by each node is no more than $\sum_{i=1}^{n} f_i(n-1)$. Moreover, we assume that OPT is not less than a small given value Δ, e.g., $\Delta = 1$. In case $OPT < \Delta$, we do not find the top-k structural hole spanners, since we are interested in users who can block a large number of information propagations rather than in those users who can block only a very small number of information propagations.

Let $T = \lceil \log_2 \frac{OPT_{ub}}{\Delta/2} \rceil$. Assume that $\frac{OPT_{ub}}{2^{s+1}} \leq OPT \leq \frac{OPT_{ub}}{2^s}$, where $0 \leq s \leq T-1$. The estimation process of OPT is as follows.

We guess OPT in decreasing order by setting $OPT_g = \frac{OPT_{ub}}{2}, \frac{OPT_{ub}}{2^2}, \ldots, \frac{OPT_{ub}}{2^t}, \ldots, \frac{OPT_{ub}}{2^T}$ one by one. For each guess $OPT_g = \frac{OPT_{ub}}{2^t}$ of OPT ($1 \leq t \leq T$), we find a set S of k nodes that block the maximum number of information propagations under $L_t = \frac{\lambda}{OPT_g} = \frac{\lambda}{OPT_{ub}} 2^t$ simulations, by invoking Algorithm 1, where $\lambda = 4(c\log n + \log kT)(2k+1)kn^2$. It can be seen that the number of simulations L_t increases exponentially with t and that $L_{t+1} = 2L_t$ for each t with $1 \leq t \leq T-1$. Recall that $\overline{H_{L_t}(S)}$ is the average number of blocked information propagations by the nodes in set S with L_t simulations. We can see that the larger L_t is, the closer the $\overline{H_{L_t}(S)}$ is to OPT.

We consider two cases. **Case 1**: The guess OPT_g is significantly larger than OPT, i.e., $OPT_g = \frac{OPT_{ub}}{2^t} \geq \frac{OPT_{ub}}{2^{s-1}}$ with $1 \leq t \leq s-1$. Then, $OPT_g = \frac{OPT_{ub}}{2^t} = 2^{s-t}\frac{OPT_{ub}}{2^s} \geq 2^{s-t} \cdot OPT$, where $2^{s-t} \geq 2^1 = 2$. We will show that it is very unlikely that the random event $\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}$ occurs, as $\frac{OPT_{ub}}{2^t}$ is considerably larger than OPT.

Case 2: The guess OPT_g is considerably smaller than OPT, i.e., $OPT_g = \frac{OPT_{ub}}{2^t} \leq \frac{OPT_{ub}}{2^{s+2}}$ with $s+2 \leq t \leq T$. Then, $OPT_g = \frac{OPT_{ub}}{2^t} = \frac{1}{2^{t-s-1}}\frac{OPT_{ub}}{2^{s+1}} \leq \frac{1}{2^{t-s-1}} OPT$, where $\frac{1}{2^{t-s-1}} \leq \frac{1}{2^1} = \frac{1}{2}$. We will prove that the event $\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}$ occurs with high probability, since $\frac{OPT_{ub}}{2^t}$ is considerably smaller than OPT.

In summary, it is very likely that the event $\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}$ occurs when $t = s, s+1$, or $s+2$. On the other hand, notice that $\frac{OPT_{ub}}{2^{s+1}} \leq OPT \leq \frac{OPT_{ub}}{2^s}$. Therefore, when the event $\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}$ occurs, the algorithm stops and sets $OPT' = \frac{1}{2}\frac{OPT_{ub}}{2^t}$. The detailed algorithm for estimating OPT is given in Algorithm 2.

Algorithm 2 Estimation of the optimal value *OPT*.

```
Input: A social network $G = (V, E)$, $f : V \mapsto \mathcal{R}^+$, $w : E \mapsto [0, 1]$, a positive integer $k$, a high probability $1 - n^{-c}$, and a small
    value $\Delta > 0$
Output: An approximate value $OPT'$ of $OPT$ so that $\frac{OPT}{8} \leq OPT' \leq OPT$ with a probability $1 - n^{-c}$ if $OPT \geq \Delta$
1:  $OPT_{ub} \leftarrow k(n-1)\sum_{i=1}^{n} f_i$; /* an upper bound on $OPT$*/
2:  $T \leftarrow \lceil \log_2 \frac{OPT_{ub}}{\Delta/2} \rceil$; /* the maximum number of iterations*/
3:  $\lambda \leftarrow 4(c\log n + \log kT)(2k+1)kn^2$;
4:  for $t \leftarrow 1$ to $T$ do
5:      Let $OPT_g = \frac{OPT_{ub}}{2^t}$; /* $OPT_g$ is a guess of $OPT$ */
6:      $L_t \leftarrow \frac{\lambda}{OPT_g}$; /* the number of simulations */
7:      Find a set $S$ of $k$ nodes that block the maximum number of information propagations under $L_t$ simulations by invoking
        Algorithm 2;
8:      if $\overline{H_{L_t}(S)} \geq OPT_g$ then
9:          $OPT' \leftarrow \frac{OPT_g}{2}$;
10:         return $OPT'$.
11:     else
12:         /* the guess $OPT_g$ of $OPT$ is too large */
13:     end if
14: end for
15: return "$OPT$ is not greater than $\Delta$".
```

3.4. Finding the top-*k* structural hole spanners

Having the $L_{\min}$, the approximation algorithm for the top-k structural hole spanner problem is given in Algorithm 3, which invokes both Algorithms 1 and 2.

Algorithm 3 maxBlock.

```
Input: A social network $G=(V,E)$, $f: V \mapsto \mathcal{R}^{+}$, $w: E \mapsto [0,1]$, a positive integer $k$, an error ratio $\epsilon$ with $0<\epsilon<1$, and a high probability $1-n^{-c}$
Output: A set $S$ of top-$k$ structural hole spanners.
1: Calculate an approximate value $OPT'$ of the optimal value $OPT$ with a high probability $1-\frac{n^{-c}}{2}$ by invoking Algorithm 2;
2: $L_{\min} \leftarrow \frac{\epsilon^{-2}((k+c)\log n+\log 4)(8k+2\epsilon)kn^2}{OPT'}$;
3: Find a set $S$ of $k$ nodes that block the maximum number of information propagations under $L_{\min}$ simulations, by invoking Algorithm 1;
4: return $S$.
```

4. Analysis of the approximation algorithm

In this section, we analyze the performance of Algorithm 3. We start by showing that the proposed estimator $\overline{H_L(u)}$ (see Eq. (3)) is unbiased for the expected number of information propagations $H(u)$ by the following lemma.

Lemma 2. *The average number of blocked information propagations $\overline{H_L(u)}$ by each node $u \in V$ in the L simulations is an unbiased estimation for its expected number $H(u)$ in G, i.e., $E[\overline{H_L(u)}]=H(u)$.*

Proof. See Appendix B. □

The rest of the section presents the analysis of the approximation ratio and time complexity of Algorithm 3 for the top-k structural hole spanner problem. We divide the analysis into two cases:

Case 1: The optimal value OPT is given, under which we show that Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with high probability if it performs $L_{OPT}=\frac{\epsilon^{-2}((k+c)\log n+\log 2)(8k+2\epsilon)kn^2}{OPT}$ simulations.

Case 2: OPT is unknown in advance; we prove that Algorithm 2 delivers an approximate value OPT' of OPT such that $\frac{OPT}{8} \le OPT' \le OPT$ with high probability. Then, the number of simulations $L=\frac{\epsilon^{-2}((k+c)\log n+\log 4)(8k+2\epsilon)kn^2}{OPT'}$ suffices for delivering a $(1-\epsilon)$-approximate solution.

4.1. Case 1: the optimal value OPT is known

Before we proceed, we first introduce the Chernoff–Hoeffding bounds [25] and the union bound, which will be used in our later analysis.

Lemma 3 (Chernoff–Hoeffding bounds)**.** *Let $X_1, X_2, \ldots, X_L$ be L independent random variables with $0 \le X_l \le 1$ and $1 \le l \le L$. Denote by $\overline{X}_L$ the average value of the L random variables and by μ the expectation of $\overline{X}_L$, i.e., $\overline{X}_L=\frac{\sum_{l=1}^{L} X_l}{L}$ and $\mu=E[\overline{X}_L]$. Then, for any constant $\delta>0$,*

$$Pr\left[\overline{X}_L-\mu \ge \delta \cdot \mu\right] \le exp\left(-\frac{\delta^2}{2+\delta}L\mu\right), \text{ and}$$

$$Pr\left[\overline{X}_L-\mu \le -\delta \cdot \mu\right] \le exp\left(-\frac{\delta^2}{2}L\mu\right).$$

Lemma 4 (The union bound)**.** *For any L random events $A_1, A_2, \ldots, A_L$, the probability that at least one of the L events happens is no more than the sum of the probabilities of the L individual events, i.e.,*

$$Pr[A_1, \text{or } A_2, \ldots, \text{or } A_L] \le \sum_{l=1}^{L} Pr[A_l]. \tag{4}$$

For any node set S with k nodes, assume that $S=\{u_1, u_2, \ldots, u_k\}$. Recall that in Algorithm 3, we approximate the expected number of blocked information propagations $H(S)(=\sum_{i=1}^{k} H(u_i))$ by nodes in S by $\overline{H_L(S)}(=\sum_{i=1}^{k}\overline{H_L(u_i)})$, where $\overline{H_L(u_i)}=n\frac{\sum_{l=1}^{L} X_l^{u_i}}{L}$, and $X_l^{u_i}$ is the number of blocked information propagations by node $u_i \in S$ in the lth simulation.

Recall that, in the lth simulations, Algorithm 3 selects a node s from the n nodes in G randomly and propagates f_s pieces of information from node s. We use $X_l^{u_i,j}$ to indicate whether the removal of node u_i blocks the information propagation from node s to a node $v_j \in V$ in the lth simulation, i.e., $X_l^{u_i,j}=1$ if the removal of node u_i blocks the information diffusion from s to v_j; otherwise, $X_l^{u_i,j}=0$. Then,

$$X_l^{u_i}=f_s \sum_{v_j \in V, v_j \ne s} X_l^{u_i,j}. \tag{5}$$

It can be seen that $0 \le X_l^{u_i} \le n-1$, as $0 \le f_s \le 1$. Let $\overline{X_l^{u_i}} = \frac{X_l^{u_i}}{n-1}$. Then, $0 \le \overline{X_l^{u_i}} \le 1$. It also can be seen that the L random variables $\overline{X_1^{u_i}}, \overline{X_2^{u_i}}, \ldots, \overline{X_L^{u_i}}$ are independently identically distributed (or i.i.d). Denote by a_i the expectation of random variable $\overline{X_l^{u_i}}$, i.e., $a_i = E[\overline{X_l^{u_i}}]$.

We can rewrite the average number $\overline{H_L(S)}$ of blocked information propagations by the nodes in S of the L simulations as

$$\overline{H_L(S)} = \sum_{i=1}^{k} \overline{H_L(u_i)} = \sum_{i=1}^{k} n \frac{\sum_{l=1}^{L} X_l^{u_i}}{L} = \sum_{i=1}^{k} n(n-1) \frac{\sum_{l=1}^{L} \overline{X_l^{u_i}}}{L}. \tag{6}$$

On the other hand, since the expectation of the random variable $\overline{H_L(S)}$ is equal to the expected number $H(S)$ of blocked information propagations by the nodes in S, we have

$$H(S) = \sum_{i=1}^{k} H(u_i) = \sum_{i=1}^{k} n(n-1)E[\overline{X_l^{u_i}}] = \sum_{i=1}^{k} n(n-1)a_i, \tag{7}$$

where $E[\overline{X_l^{u_i}}] = a_i$. Then, $a_i = \frac{H(u_i)}{n(n-1)}$. Let $a_S = \sum_{u_i \in S} a_i$. Then, $a_S = \frac{H(S)}{n(n-1)}$.

Proof roadmap: In the following, we first analyze the upper bounds on $Pr[\overline{H_L(S)} - H(S) \ge \epsilon \cdot H(S)]$ and $Pr[\overline{H_L(S)} - H(S) \le -\epsilon \cdot H(S)]$ in Lemma 5. We then show the lower bound on $Pr[|\overline{H_L(S)} - H(S)| \le \frac{\epsilon}{2} OPT]$ in Lemma 6, where $Pr[X]$ is the probability t-hat an event X occurs. Finally, we prove that Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with a high probability $1 - n^{-c}$ if $L \ge \frac{\epsilon^{-2}((k+c)\log n + \log 2)(8k+2\epsilon)kn^2}{OPT}$ in Lemma 7, where $c \ge 1$.

We start by analyzing the upper bounds on $Pr[\overline{H_L(S)} - H(S) \ge \epsilon \cdot H(S)]$ and $Pr[\overline{H_L(S)} - H(S) \le -\epsilon \cdot H(S)]$ in Lemma 5.

Lemma 5. *For any k-size set $S = \{u_1, u_2, \ldots, u_k\}$, let $\overline{X_1^{u_i}}, \overline{X_2^{u_i}}, \ldots, \overline{X_L^{u_i}}$ be independent variables for each $u_i \in S$ with $0 \le \overline{X_l^{u_i}} \le 1$. Given any constant ϵ with $0 < \epsilon < 1$,*

$$Pr\left[\overline{H_L(S)} - H(S) \ge \epsilon \cdot H(S)\right] \le k \cdot exp\left(-\frac{\epsilon^2 L}{(2k+\epsilon)kn^2} H(S)\right), \tag{8}$$

$$Pr\left[\overline{H_L(S)} - H(S) \le -\epsilon \cdot H(S)\right] \le k \cdot exp\left(-\frac{\epsilon^2 L}{2k^2 n^2} H(S)\right). \tag{9}$$

Proof. See Appendix C. □

We then show the lower bound on $Pr[|\overline{H_L(S)} - H(S)| \le \frac{\epsilon}{2} OPT]$ in Lemma 6.

Lemma 6. *For any set S with k nodes, let $\overline{H_L(S)}$ be the average number of blocked information propagations by the nodes in S derived from L simulations. Given an error ratio ϵ with $0 < \epsilon < 1$ and a positive constant $c \ge 1$, we have*

$$Pr\left[|\overline{H_L(S)} - H(S)| \le \frac{\epsilon}{2} OPT\right] \ge 1 - \frac{n^{-c}}{\binom{n}{k}}, \text{ if} \tag{10}$$

$$L \ge \frac{\epsilon^{-2}((k+c)\log n + \log 2)(8k+2\epsilon)kn^2}{OPT}, \tag{11}$$

where $\binom{n}{k}$ is the number of ways of choosing k nodes from a set of n nodes.

Proof. See Appendix D. □

Let S^* be an optimal solution, i.e., $OPT = H(S^*) = \max_{S \subset V, |S|=k}\{H(S)\}$. We show that Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with high probability if $L \ge \frac{\epsilon^{-2}((k+c)\log n + \log 2)(8k+2\epsilon)kn^2}{OPT}$ in the following lemma.

Lemma 7. *Given a constant ϵ with $0 < \epsilon < 1$ and a high probability $1 - n^{-c}$, Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with probability $1 - n^{-c}$ if $L \ge \frac{\epsilon^{-2}((k+c)\log n + \log 2)(8k+2\epsilon)kn^2}{OPT}$.*

Proof. See Appendix E. □

4.2. Case 2: the optimal value OPT is unknown

Thus far, we have assumed that the optimal value OPT is given; we now remove this assumption by calculating an approximate value OPT' of OPT such that $\frac{OPT}{8} \le OPT' \le OPT$ with high probability by invoking Algorithm 2. We start with two important observations 1 and 2, which will be used in our later analysis.

Observation 1. Given a random variable X defined in an interval I, then for any two numbers $x, y \in I$ with $x \geq y$,

$$Pr[X \geq x] \leq Pr[X \geq y]. \tag{12}$$

Proof. We can see that $Pr[X < y] \leq Pr[X < x]$ as $y \leq x$. Then, $Pr[X \geq x] = 1 - Pr[X < x] \leq 1 - Pr[X < y] = Pr[X \geq y]$. □

Observation 2. Given two random variables X and Y defined in an interval I with $X \geq Y$, then for any number $x \in I$,

$$Pr[X \leq x] \leq Pr[Y \leq x]. \tag{13}$$

Proof. It can be seen that the occurrence of the random event $X \leq x$ implies that the event $Y \leq x$ also happens since $Y \leq X \leq x$. The lemma then follows. □

Recall that, in Algorithm 2, it guesses the optimal value OPT in decreasing order with $OPT_g = \frac{OPT_{ub}}{2}, \frac{OPT_{ub}}{2^2}, \ldots, \frac{OPT_{ub}}{2^t}, \ldots, \frac{OPT_{ub}}{2^T}$. Given a guess $OPT_g = \frac{OPT_{ub}}{2^t}$ of OPT $(1 \leq t \leq T)$, it finds a set S of k nodes that block the maximum number of information propagations under $L_t = \frac{\lambda}{OPT_g}$ simulations.

Proof roadmap: In the following, we show that it is very unlikely that the random event $\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}$ occurs when $1 \leq t \leq s-1$ in Lemma 8, although it almost surely happens when $s+2 \leq t \leq T$ in Lemma 9. Then, it is very likely that Algorithm 2 terminates (i.e., the event $\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}$ happens), when $t = s, s+1$, or $s+2$. Therefore, the delivered approximate value OPT' falls in the interval $[\frac{OPT}{8}, OPT]$ with high probability, as $\frac{OPT_{ub}}{2^{s+1}} \leq OPT \leq \frac{OPT_{ub}}{2^s}$, as shown in Lemma 10. Finally, by substituting OPT' for OPT, we show that Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with high probability in Theorem 1.

We start by showing the upper bound on $Pr[\overline{H_{L_t}(S)} \geq OPT_g]$ when $1 \leq t \leq s-1$ by Lemma 8.

Lemma 8. *Given a guess OPT_g of OPT with $OPT_g = \frac{OPT_{ub}}{2^t} \geq \frac{OPT_{ub}}{2^{s-1}}$, then*

$$Pr\left[\overline{H_{L_t}(S)} \geq OPT_g\right] \leq \frac{n^{-c}}{T}, \tag{14}$$

where $1 \leq t \leq s-1$.

Proof. See Appendix F. □

Consider the solution S found within L_t simulations. Denote by S^* an optimal solution to the problem. We then show the lower bound on $Pr[\overline{H_{L_t}(S)} < OPT_g]$ when $s+2 \leq t \leq T$ by Lemma 9.

Lemma 9. *Given a guess OPT_g of OPT with $OPT_g = \frac{OPT_{ub}}{2^t} \leq \frac{OPT_{ub}}{2^{s+2}}$, then*

$$Pr\left[\overline{H_{L_t}(S)} \geq OPT_g\right] \geq 1 - \frac{n^{-c}}{2^{t-s-1}T}, \tag{15}$$

where $s+2 \leq t \leq T$.

Proof. See Appendix G. □

We show that it is very likely that the random event $\frac{OPT}{8} \leq OPT' \leq OPT$ occurs by the following Lemma.

Lemma 10. *Algorithm 2 delivers an approximate value OPT' of the optimal one OPT, such that $\frac{OPT}{8} \leq OPT' \leq OPT$ with a high probability of not less than $1 - n^{-c}$. Moreover, the expected time of Algorithm 2 is $O(\frac{k^2 n^2 m(\log k + c\log n)}{OPT})$.*

Proof. See Appendix H. □

Finally, we show that Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with high probability.

Theorem 1. *Given a social network $G = (V, E)$, $f: V \mapsto \mathcal{R}^+$, w: $E \mapsto [0, 1]$, and two constants ϵ and c with $0 < \epsilon < 1$ and $c \geq 1$, respectively, Algorithm 3 delivers a $(1-\epsilon)$-approximate solution to the top-k structural hole spanner problem with a high probability $1 - n^{-c}$ in an expected time of $O(\frac{\epsilon^{-2}(k+c)k^2 n^2 m \log n}{OPT})$ if $OPT \geq \Delta$, where $n = |V|$, $m = |E|$, Δ is a given small value with $\Delta \geq 1$, and OPT is the optimal value.*

Proof. Following Lemma 10, Algorithm 2 can deliver an approximate value OPT' of OPT with $\frac{OPT}{8} \leq OPT' \leq OPT$ with a probability $1 - \frac{n^{-c}}{2}$ in an expected time of $O(\frac{k^2 n^2 m(\log k + c\log n)}{OPT})$. Having the approximate value OPT', Algorithm 3 then delivers a $(1-\epsilon)$-approximate solution with a high probability $1 - \frac{n^{-c}}{2}$ by performing $L = \frac{\epsilon^{-2}((k+c)\log n + \log 4)(8k+2\epsilon)kn^2}{OPT'}$ simulations, following Lemma 7. The expected time of Algorithm 3 is thus $O(\frac{\epsilon^{-2}(k+c)k^2 n^2 m \log n}{OPT})$. Moreover, it can be seen that Algorithm 3 delivers a $(1-\epsilon)$-approximate solution with probability $1 - 2\frac{n^{-c}}{2} = 1 - n^{-c}$ by the union bound. □

5. A fast heuristic algorithm

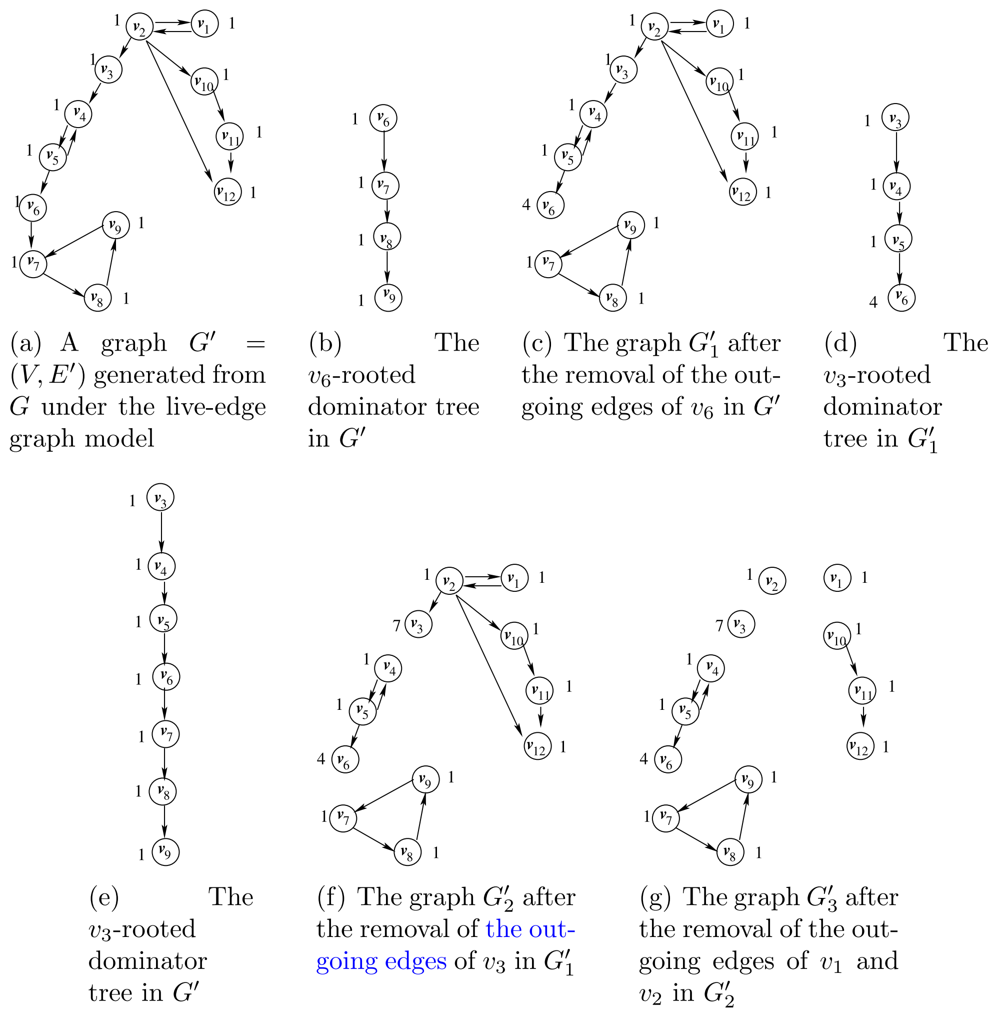
Although the approximate solution delivered in the previous section provides a guaranteed performance, it takes a considerably long time to deliver the solution (see Theorem 1), especially when the size of a social network is very large, e.g., millions of nodes. In this section, we present a fast, yet scalable, heuristic algorithm for the problem.

5.1. Basic idea of the proposed algorithm

Before we proceed, we define the necessary notations as follows. Given a directed graph $G' = (V, E')$, for any two different nodes u and v in G', if there is a directed path from u to v in G', then u is referred to as an *ancestor* of v, and v is referred to as a *descendant* of u. A node v in G' is referred to as a *separation node* if and only if node v must be contained in any path from each ancestor v_A of v to each descendant v_D of v in G'. In other words, the removal of v from G' will disconnect its ancestors from its descendants. For example, in Fig. 4(a), only nodes v_3 and v_6 are separation nodes in G'. In contrast, the other nodes are not separation nodes. For example, v_4 is not a separation node, since v_5 is both the ancestor and the descendent of v_4 and v_5 is reachable from itself.

Recall that, within each of the L simulations in the approximation algorithm (see Algorithm 1), it considers the information propagation from only one source node s to every other node, and computes the number of blocked information propagations by each node. Unlike the approximation algorithm, the heuristic algorithm here considers the information propagations from the $n(= |V|)$ source nodes simultaneously in a novel way, thereby reducing the number of simulations.

Like in the approximation algorithm, within each simulation, the heuristic algorithm first generates a directed graph $G' = (V, E')$ from G randomly under the live-edge graph model. To efficiently compute the number of blocked information



(a) A graph $G' = (V, E')$ generated from G under the live-edge graph model

(b) The v_6-rooted dominator tree in G'

(c) The graph G'_1 after the removal of the outgoing edges of v_6 in G'

(d) The v_3-rooted dominator tree in G'_1

(e) The v_3-rooted dominator tree in G'

(f) The graph G'_2 after the removal of the outgoing edges of v_3 in G'_1

(g) The graph G'_3 after the removal of the outgoing edges of v_1 and v_2 in G'_2

Fig. 4. Illustration of the heuristic algorithm.

propagations by each node from the n source nodes in G', the algorithm exploits two important properties of a separation node v.

Property (i): For each descendant v_D of v in G', if the removal of node v_D blocks a piece of information propagation from node v to another descendent v'_D of v, then the removal also blocks an information propagation from each its ancestor v_A to node v'_D. For example, consider the separation node v_6 in Fig. 4(a). Both nodes v_7 and v_8 are its descendants, and the removal of v_7 blocks an information diffusion from v_6 to v_8. Then, the removal of v_7 also blocks an information diffusion from any ancestor (e.g., v_5) of v_6 to v_8.

We illustrate how this property facilitates the efficient computation of the blocked information propagations. Consider the v_6-rooted dominator tree in G' (see Fig. 4(b)). It can be seen that the removal of a dominator node, e.g., v_7, will not only blocks an information propagation from the tree root v_6 to the two descendants of v_7 (i.e., v_8 and v_9), but also will block an information propagation from each ancestor of v_6 in G', i.e., v_1, v_2, v_3, v_4, v_5, to the two descendants. Then, the number of blocked information propagations by v_7 is $\sum_{i=1}^{6} f_i \cdot 2$, where f_i is the information sharing frequency of node v_i and the number of descendants of v_7 in the v_6-rooted dominator tree is 2. Once the information propagations from node v_6 have been considered, the outgoing edges of this node can be removed from G' (see Fig. 4(c)).

Property (ii): For each ancestor of v_A of the separation node v, if the removal of v_A blocks an information propagation from its other ancestor v'_A to v, then this removal also blocks an information propagation from v'_A to each descendant v_D to v. For instance, both nodes v_4 and v_5 are ancestors of v_6 and the removal of v_5 blocks an information propagation from v_4 to v_6. Then, the removal of v_5 also blocks an information diffusion from v_4 to any descendant (e.g., v_7) of v_6. Therefore, we associate a weight with each node, which is the sum of the numbers of its descendants in G' and 1 (i.e., itself). For example, Fig. 4(c) shows the graph G'_1 after the removal of the outgoing edges of v_6 in G' and the weight of v_6 in G'_1 is increased from 1 to 4.

From the example in Fig. 4, it can be seen that the removal of the outgoing edges of separation node v_6 splits G' into two smaller connected subgraphs in G'_1, and the number of edges in graph G'_1 is smaller than that in graph G'. The edge removal will thus save time for computing the number of blocked information propagations from the ancestors of separation node v_6. For instance, Fig. 4(d) illustrates the dominator tree rooted at v_3 in G'_1, and its size is smaller than that of the v_3-rooted dominator tree in G', as illustrated in Fig. 4(e).

5.2. Algorithm

The heuristic algorithm performs L Monte Carlo simulations and calculates the average number of blocked information propagations by each node in the L simulations. The top-k structural hole spanners in G then are the k nodes that block the maximum numbers of information propagations, where L is a given positive number.

Within each of the L simulations, the algorithm first generates a graph $G' = (V, E')$ from G under the live-edge graph model. It then performs n_s iterations to calculate the number of blocked information propagations from the n source nodes by each node, where $n = |V|$ and n_s is the number of *strongly connected components (SCCs)* in G'. Denote by G'_{p-1} and G'_p the graphs before and after the pth iteration, respectively, where $1 \le p \le n_s$. Moreover, we associate each node v_i in G'_p with a positive number d_i to represent the sum of the number of its descendants and 1 (i.e., itself). Initially, $G'_0 = G'$ and $d_i = 1$ for each $v_i \in G'_0$, e.g., see Fig. 4(a).

Consider the pth iteration with $1 \le p \le n_s$. We first identify all separation nodes in $G'_{p-1} = (V, E'_{p-1})$, and such an identification will be shown in the next subsection. Then, only nodes v_3 and v_6 are separation nodes in G' of Fig. 4(a).

Having identified the separation nodes in G'_{p-1}, we then consider the information propagations from some nodes by dividing them into the following two cases.

Case 1: There is at least one separation node in G'_{p-1}, we choose a separation node v such that it is not an ancestor of any other separation node. For example, in Fig. 4(a), although both nodes v_3 and v_6 are separation nodes, we choose node v_6 since v_3 is an ancestor of v_6.

We then construct a v-rooted dominator tree T_v by invoking the algorithm in [11], e.g., Fig. 4(b). Denote by V_v^A the set of ancestors of v in G'_{p-1}. Moreover, for each $u \in T_v$, denote by D_u the weighted sum of its descendants in T_v, i.e., $D_u = \sum_{v_j \in R_{T_v}(u) \setminus \{u\}} d_j$, where $R_{T_v}(u) \setminus \{u\}$ is the set of proper descendants of node u in the dominator tree T_v, and d_j is the weight of node v_j. We calculate the number of blocked information propagations by each node u in T_v as $(f_v + \sum_{v_i \in V_v^A} f_i) \cdot D_u$ if $u \neq v$; otherwise $(u = v)$, the number is $(\sum_{v_i \in V_v^A} f_i) \cdot D_u$, where f_i is the information sharing frequency of node v_i. For example, in Fig. 4(b), the number of blocked information propagations by node v_7 is $\sum_{i=1}^{6} f_i \cdot 2$, while that by v_6 is $\sum_{i=1}^{5} f_i \cdot 3$. After the calculation, we obtain graph G'_p from G'_{p-1} by removing the outgoing edges of node v from G'_{p-1} and updating the weight d_v of node v as $d_v = \sum_{v_j \in T_v} d_j$. For example, the weight d_6 of node v_6 is updated as $d_6 = 4$, see Fig. 4(c).

Case 2: There are no separation nodes in G'_{p-1}, we choose a set V' of nodes in the same SCC, such that they have no incoming neighbors outside the SCC in G'_{p-1}, and they have not been considered before, e.g., the SCC consisting of nodes v_1 and v_2 in Fig. 4(f). Then, for each node v in set V', we calculate the number of blocked information propagations from v by each node by invoking the algorithm in [11]. After the calculations with these nodes, we finally obtain graph G'_p from G'_{p-1} by removing the outgoing edges of nodes in V'.

The reason for the choice of the set of nodes in the same SCC without any incoming neighbors outside the SCC is that there may be new separation nodes in G'_p after the edge removals. For example, after the removals of the outgoing edges of nodes v_1 and v_2 in G'_2 of Fig. 4(f), node v_{11} becomes a new separation node in graph G'_3 (see Fig. 4(g)).

The heuristic algorithm is presented in Algorithm 4.

Algorithm 4 maxBlockFast.

```
Input: A social network $G = (V, E)$, $f : V \mapsto \mathcal{R}^+$, $w : E \mapsto [0, 1]$, an integer $k > 0$, and the number of simulations $L$.
Output: A set $S$ of $k$ nodes that block the maximum information propagations under $L$ simulations
1:  Let $h_u \leftarrow 0$ for each $u \in V$;
2:  for $l \leftarrow 1$ to $L$ do
3:    Generate a graph $G' = (V, E')$ from $G$ under the live-edge graph model.
4:    Let $G'_0(V, E'_0) = G'$, $d_i = 1$ for each $v_i \in V$, and $n_s$ be the number of SCCs in $G'_0$;
5:    for $p \leftarrow 1$ to $n_s$ do
6:      Find separation nodes in $G'_{p-1}$;
7:      if there are separation nodes in $G'_{p-1}$ then
8:        Choose a separation node $v$ that is not an ancestor of any other separation node;
9:        Construct a $v$-rooted dominator tree $T_v$ in $G'_{p-1}$ by invoking the algorithm in [11];
10:       For each node $u \in T_v$, let $D_u = \sum_{v_j \in R_{T_v}(u) \setminus \{u\}} d_j$, and $h_u \leftarrow h_u + (f_v + \sum_{v_i \in V_v^A} f_i) \cdot D_u$ if $u \neq v$; otherwise $(u = v)$, let
          $h_u \leftarrow h_u + \sum_{v_i \in V_v^A} f_i \cdot D_u$;
11:       Let $d_v = \sum_{v_j \in T_v} d_j$;
12:       Obtain graph $G'_p$ from $G'_{p-1}$ by removing the outgoing edges of $v$;
13:     else
14:       Choose a set $V'$ of nodes in the same SCC in $G'_{p-1}$, such that they have no incoming neighbors outside the SCC
          and they have not been considered before;
15:       for each node $v$ in $V'$ do
16:         Construct a $v$-rooted dominator tree $T_v$ in $G_{p-1}$ by invoking the algorithm in [11];
17:         For each node $u \in T_v$, let $h_u \leftarrow h_u + f_v \cdot \sum_{v_j \in R_{T_v}(u) \setminus \{u\}} d_j$;
18:       end for
19:       Obtain graph $G'_p$ from $G'_{p-1}$ by removing the outgoing edges of nodes in $V'$;
20:     end if
21:   end for
22: end for
23: Let $\overline{H_L(u)} \leftarrow \frac{h_u}{L}$ for each node $u \in V$;
24: Sort the nodes in $V$ in nonincreasing order by their values $\overline{H_L(u)}$ and assume that $\overline{H_L(u_1)} \geq \overline{H_L(u_2)} \geq \cdots \geq \overline{H_L(u_n)}$;
25: return $S = \{u_1, u_2, \ldots, u_k\}$.
```

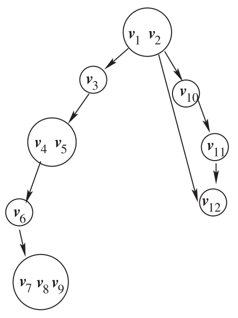
5.3. *Efficient identification of separation nodes*

We now describe how to efficiently find all separation nodes in graph G'_{p-1}. Assume that there are n_s strongly connected components $SCC_1, SCC_2, \ldots, SCC_{n_s}$ in G'_{p-1}. We first construct a graph $G_s = (V_s, E_s)$ from G'_{p-1}, where each *supernode*$u_i^s \in V_s$ represents the SCC_i in G'_{p-1}, and there is an edge from u_i^s to u_j^s in E_s if there is an edge from a node in SCC_i to another node in SCC_j in G'_{p-1} with $i \neq j$. Fig. 5(a) shows the constructed graph G_s from G' in Fig. 4(a) by collapsing each strongly connected component in G' into a supernode in G_s. It can be seen that G_s is a *Directed Acyclic Graph (DAG)*.

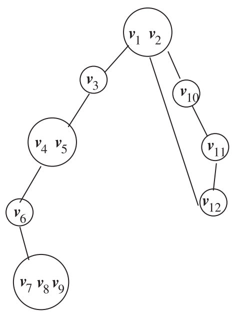
Notice that, for each separation node v in G'_{p-1}, the strongly connected component SCC_i in which v is contained must contain node v only, and Fig. 5(a) shows the strongly connected component in which node v_6 is contained. Otherwise (SCC_i consists of at least two nodes, e.g., nodes u and v), assume that node v is a separation node. Since nodes u and v are in the same SCC, node u is not only an ancestor of v but also a descendant of v. The removal of node v cannot disconnect its ancestors and descendants, since node u is reachable from itself.

It can be seen that node v is a separation node in G'_{p-1} if and only if SCC_i consists of node v only and the supernode u_i^s that represents SCC_i is also a separation node in G_s. We find all separation nodes in G_s as follows.

We consider the *undirected underlying graph* $G'_s = (V'_s, E'_s)$ of G_s by replacing each directed edge in G_s with an undirected edge (see Fig. 5(b)). Assume that each node $u^s \in G'_s$ has d_u neighbors $v_1^s, v_2^s, \ldots, v_{d_u}^s$ with $d_u \geq 0$. More, assume that u^s is contained in a connected component CC_u in G'_s and that the removal of u^s disconnects CC_u into n_u connected components $CC'_1, CC'_2, \ldots, CC'_{n_u}$, where $n_u \geq 1$. If $n_u \geq 2$, node u^s is then referred to as a *cut node* in G'_s. On the other hand, the removal of u^s also partitions its d_u neighbors into n_u groups $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_{n_u}$, where those neighbors are in the same group $\mathcal{G}_j$ if they are in the same connected component CC'_j.



(a) The constructed graph G_s from G' in Fig. 4(a)



(b) The undirected underlying graph G'_s of G_s

Fig. 5. Illustration of the process of finding separation nodes.

We can see that a separation node u^s in G_s must be a cut node in G'_s. However, not every cut node in G'_s is a separation node in G_s. The sufficient and necessary conditions of a separation node are shown by the following lemma.

Lemma 11. *A node u^s is a separation node in G_s if and only if the following three conditions are met:*

(i) *Node u^s has at least one incoming neighbor and one outgoing neighbor in G_s.*
(ii) *Node u^s is a cut node in the underlying undirected graph G'_s.*
(iii) *For each group $\mathcal{G}_j$ with $1 \le j \le n_u$, all neighbors of u^s in $\mathcal{G}_j$ are either incoming neighbors or outgoing neighbors of u^s in G_s, but not both. That is, there are no two neighbors v^s_1 and v^s_2 in $\mathcal{G}_j$ such that v^s_1 is an incoming neighbor of u^s and v^s_2 is an outgoing neighbor of u^s.*

The proof is given in Appendix I.

Notice that we can find all cut nodes in G'_s by performing only one depth-first-search on G'_s [34].

Theorem 2. *Given a social network $G = (V, E)$, $f : V \mapsto \mathcal{R}^+$, w: $E \mapsto [0, 1]$, and the number of simulations L, there is a heuristic algorithm, Algorithm 4, for the top-k structural hole spanner problem in G with a running time of $O(Ln(n + m))$, where $n = |V|$ and $m = |E|$.*

Proof. The proof is straightforward and thus omitted. □

6. Performance evaluation

In this section, we evaluate the performance of the proposed algorithms, using real-world datasets.

6.1. Experimental setting

Dataset description: We adopted seven real-world network datasets listed in Table 1, where the first dataset, `GR-QC`, is the collaboration network from arXiv[1], representing the collaboration relationships of authors of papers submitted to the General Relativity and Quantum Cosmology category. The second dataset is from the online social network `Facebook`[2]. The third dataset `Twitter` was obtained from [23]. The fourth dataset `Email-EuAll` is the anonymous email network

Table 1
Statistics of seven real-world networks.

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	Average degree
GR-QC	5244	28,968	5.5
Facebook	63,731	1,634,180	25.6
Twitter	92,180	377,942	4.1
Email-euAll	265,214	728,962	2.7
DBLP-2011	986,324	6,707,236	6.8
LiveJournal	5,363,260	54,880,888	10.2
Coauthor	53,442	255,936	4.8

[1] http://arxiv.org/.
[2] http://socialnetworks.mpi-sws.org/data-wosn2009.html.

of a large European research institution for an 18-month period [33]. The fifth dataset DBLP-2011 is the collaboration network obtained from the DBLP web site[3]. The sixth dataset LiveJournal describes the social network of free online blogging community.[4] The final dataset Coauthor was also obtained from [23], which not only consists of authors and their co-author relationships, but also contains ground-truth communities, where the communities are publication venues, e.g., journals or conferences, and authors who have published in the same journal or conference form a community.

For each of the seven networks, the information sharing frequency f_i of each user v_i was randomly chosen from an interval $[0, f_{\max}]$ with $f_{\max} = 1$ piece of information per day, and the activation probability $w_{i,j}$ of each edge was randomly selected from an interval $[0, w_{\max}]$ with $w_{\max} = 0.1$.

The number k of to-be-identified structural hole spanners varied from 1 to 50. For the proposed algorithm maxBlock, the default error ratio was $\epsilon = 0.5$ and the approximation ratio of maxBlock then was $1 - \epsilon = 0.5$. Algorithm maxBlock runs with a high probability of $1 - n^{-c} = 1 - \frac{1}{n}$ when $c = 1$. Following Theorem 1, more simulations are needed for a larger social network G. Therefore, the number of simulations L in algorithm maxBlockFast was set to $L = \log_2 n$, where n is the number of users in G. All experiments were performed on a server with an Intel(R) Core(TM) i7-4790 CPU (3.6 GHz) and 8 GB RAM.

Table 2 lists the parameters used in all experiments.

Table 2
Parameters used in the experiments.

Parameters	Values
Information sharing frequency f_i	[0, 1]
Activation probability $w_{i,j}$	[0, 0.1]
Number of structural holes k	[1, 50]
Approximation ratio $1 - \epsilon$ of maxBlock	1-0.5=0.5
Success probability $1 - n^{-c}$ of maxBlock	$1 - \frac{1}{n}$
Simulation time L of maxBlockFast	$\log_2 n$

Benchmark algorithms: To evaluate the performance of the proposed algorithms maxBlock and maxBlockFast, we compared them against the following nine existing algorithms.

(1) Algorithm PathCount [14] counts the number of shortest paths on which each node lies, among all-pairs shortest paths, and chooses the k nodes with the largest numbers.

(2) Algorithm 2Step [29] computes the number of shortest paths with a length of just two, on which each node lies, and then selects the k nodes with the top-k largest numbers.

(3) Algorithm PageRank [26] calculates the visiting probability $r(v)$ of each node v. The algorithm computes $r(v)$ iteratively. Initially, $r_0(v) = \frac{1}{n}$. The algorithm then updates $r_{t+1}(v)$ by $r_{t+1}(v) = (1-\alpha)/n + \alpha \sum_{(u,v)\in E} r_t(u)/d_u$, where $\alpha = 0.85$ and d_u is the degree of node u. It finally chooses the top-k nodes with the highest visiting probabilities when the algorithm converges.

(4) Algorithm Constraint [3] measures the constraint on every user by his friends and selects k nodes with the lowest constraints.

(5) Algorithm AP-Greedy [34] chooses the top-k nodes so that the increased communication cost of the residual network after their removal is maximized.

(6) Algorithm maxInfluence [30] identifies a set of k nodes with the maximum influence.

(7) Algorithm HAM [16] proposes a harmonic modularity method to jointly detect communities and hole spanners simultaneously.

(8) Algorithm HIS [23] jointly computes the importance of each node v to its communities and the extent to which node v bridges the communities, assuming that communities are given.

(9) Algorithm MaxD [23] finds a set of k nodes such that the minimum cut of the communities will be decreased significantly after the removal of the k nodes.

Notice that algorithms HAM, HIS, and MaxD were not applicable to all of the seven networks owing to the following reasons.

Algorithm HAM [16] was evaluated only for the smallest network GR-QC with 5244 nodes, because its space complexity is very high, i.e., $O(n^2)$ [16], while the space complexities of all the other algorithms are only $O(n+m)$. For example, the memory consumption of HAM for network GR-QC was about 1 GB in our experiments and its space consumption for the second smallest network Coauthor with about 53,000 nodes was expected to grow up to 100 GB. On the other hand, algorithms HIS and MaxD [23] were only applicable to the Coauthor network with its information about ground-truth communities, as both algorithms have a prerequisite, i.e., all communities must be given in advance. However, it is worth noting that ground-truth communities in the other six networks are not available.

[3] http://dblp.uni-trier.de/.
[4] http://livejournal.com/.

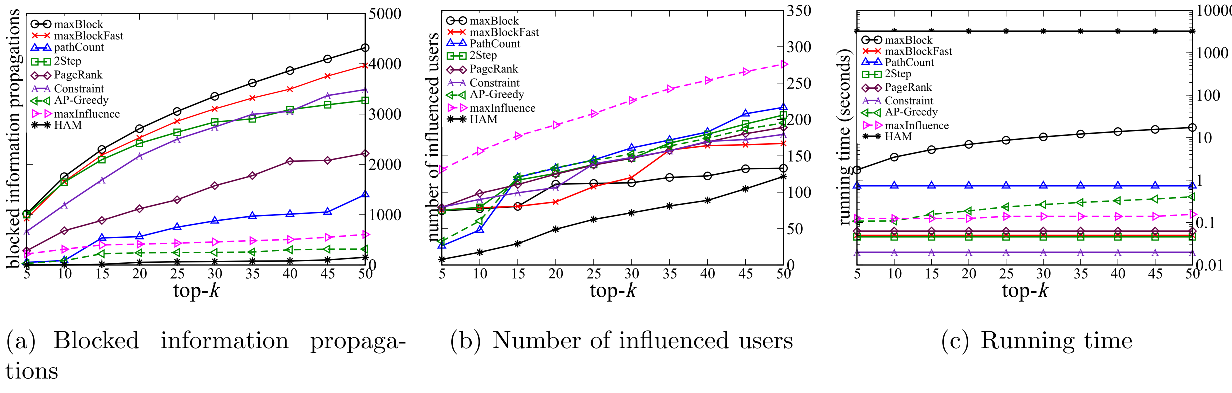


Fig. 6. Performance of the algorithms in the GR-QC network.

6.2. Algorithm performance evaluation

We first compared the performance of the proposed algorithms maxBlock and maxBlockFast against algorithms PathCount, 2Step, PageRank, Constraint, AP-Greedy, maxInfluence, and HAM in the GR-QC network. Fig. 6(a) shows that the number of blocked information propagations by the removal of the identified hole spanners by each of the algorithms increases with the growth of k, and maxBlock outperforms all the other mentioned algorithms. For example, the number of blocked information propagations by maxBlock is 24% ($\approx \frac{4{,}320-3{,}486}{3{,}486}$) larger than that by the best-existing algorithm (i.e., Constraint) when $k = 50$, and the number by the proposed heuristic algorithm maxBlockFast is approximately 8% less than the number by maxBlock, where the numbers of blocked information propagations by maxBlock, maxBlockFast, PathCount, 2Step, PageRank, Constraint, AP-Greedy, maxInfluence, and HAM are 4,320, 3,964, 1,399, 3,271, 2,217, 3,486, 315, 606, and 153, respectively. Fig. 6(a) also shows that the number of blocked propagations by maxInfluence is very small, e.g., only 606 information propagations when $k = 50$, even though the nodes identified by it influence the maximum expected number of users (see Fig. 6(b)). This shows that Fig. 6(a) and (b) validates our claim that the influence maximization problem is essentially different from the top-k structural hole spanner problem in this paper. Fig. 6(c) plots the running times of the nine algorithms, from which it can be seen that HAM has the longest running time, i.e., more than 3000 s, and maxBlock has a running time of approximately 17 seconds, while the proposed heuristic algorithm maxBlockFast has a much shorter running time, with approximately 0.05 seconds, and the other algorithms have running times not longer than 1 second.

We then studied the algorithm performance in the Facebook network, whose size is bigger than that of the GR-QC network. Fig. 7(a) demonstrates that the numbers of blocked information propagations by maxBlock and maxBlockFast are still the maximum ones, which are at least 10% larger than those by PathCount, 2Step, PageRank, or Constraint, and are significantly larger than those by both AP-Greedy and maxInfluence. For example, the numbers of blocked information propagations by maxBlock, maxBlockFast, PathCount, 2Step, PageRank, and Constraint are 1.0×10^7,

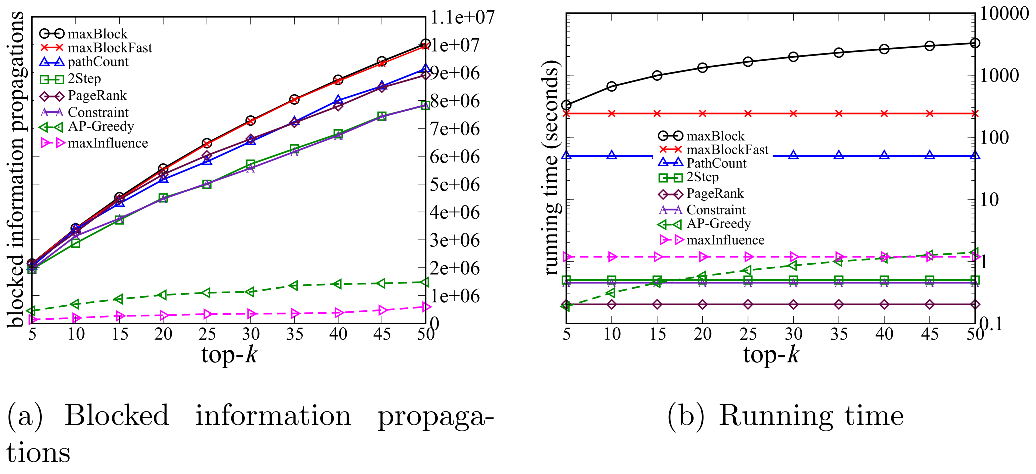
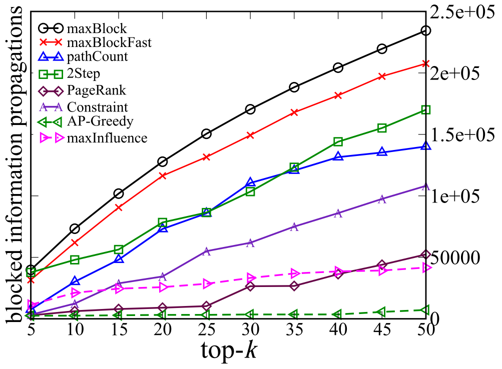
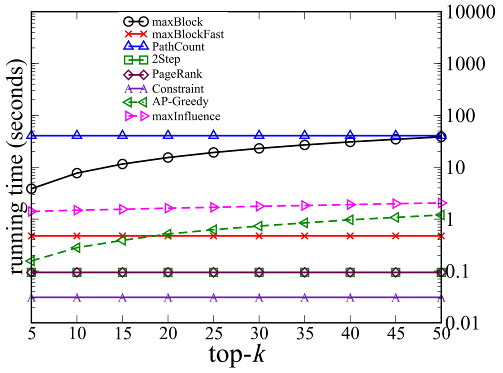


Fig. 7. Performance of the algorithms in the Facebook network.



(a) Blocked information propagations



(b) Running time

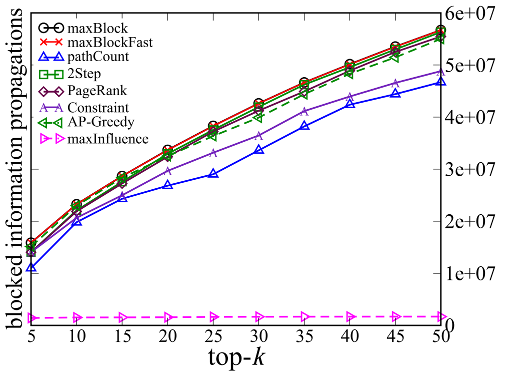
Fig. 8. Performance of the algorithms in the `Twitter` network.

9.95×10^6, 9.1×10^6, 7.8×10^6, 8.9×10^6, and 7.8×10^6, respectively when $k = 50$. Moreover, Fig. 7(b) shows that the running times of `maxBlock`, `maxBlockFast`, and `PathCount` are much longer than those of the other algorithms, which are about 3300, 240, and 50 s, respectively, while the running times of the other algorithms are no more than 2 seconds. This reflects a non-trivial trade-off between the solution quality and the running time for delivering the solution.

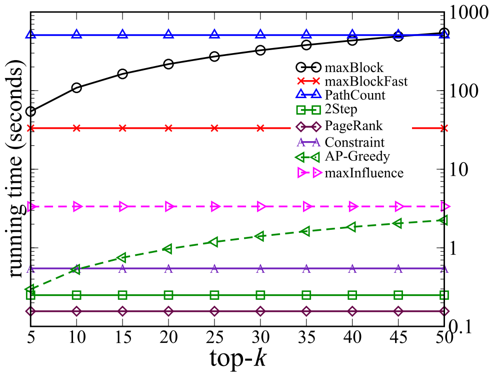
We now investigate the performance of the different algorithms in the `Twitter` network. Fig. 8(a) clearly shows that the removals of the identified hole spanners by both `maxBlock` and `maxBlockFast` block much more information propagations than those by the other algorithms. For instance, the removals of the top 50 nodes found by these two block more than 38% ($\approx \frac{234,480-170,000}{170,000}$) and 22% ($\approx \frac{20,7600-170,000}{170,000}$) information propagations compared to that by the best-existing algorithm `2Step`, respectively. Fig. 8(b) plots that the running times of `maxBlock` and `maxBlockFast` are approximately 40 and 0.5 s, respectively.

We further studied the performance of the different algorithms in the `Email-euAll` network. Fig. 9(a) shows that the numbers of blocked information propagations by `maxBlock` and `maxBlockFast` are only slightly larger than those by `2Step`, `PageRank`, and `AP-Greedy`, approximately 16% more than those by `Constraint` and `PathCount`, and even 33 times the number by `maxInfluence`, when $k = 50$. Moreover, Fig. 9(b) shows that the running times of `maxBlock`, `maxBlockFast`, and `PathCount` are about 550, 35, and 510 s, respectively, while the running times of the other five algorithms are not longer than 5 s.

We evaluated the algorithm performance in the `DBLP-11` network. Fig. 10(a) demonstrates that the numbers of blocked information propagations by `maxBlock` and `maxBlockFast` are slightly more than those by `PathCount`, `2Step`, `PageRank`, and `Constraint`, but are at least three times the numbers by `AP-Greedy` and `maxInfluence`. On the other

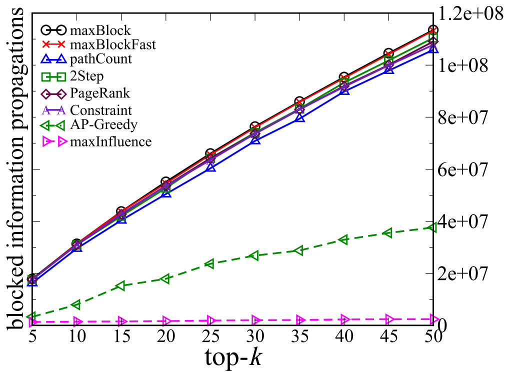


(a) Blocked information propagations

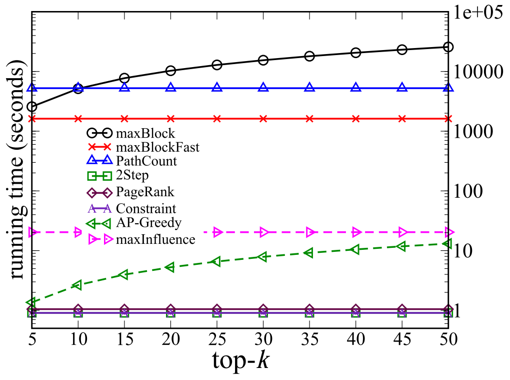


(b) Running time

Fig. 9. Performance of the algorithms in the `Email-euAll` network.

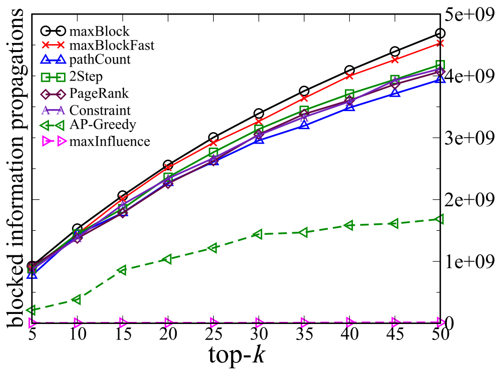


(a) Blocked information propagations

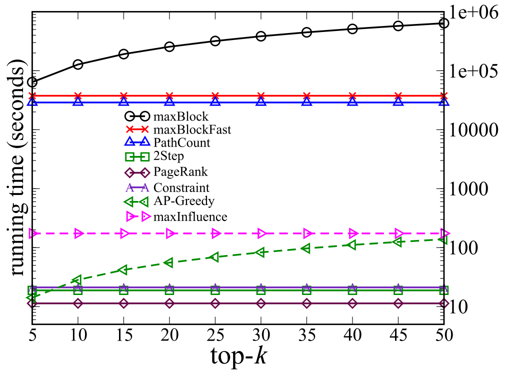


(b) Running time

Fig. 10. Performance of the algorithms in the DBLP-11 network.



(a) Blocked information propagations



(b) Running time

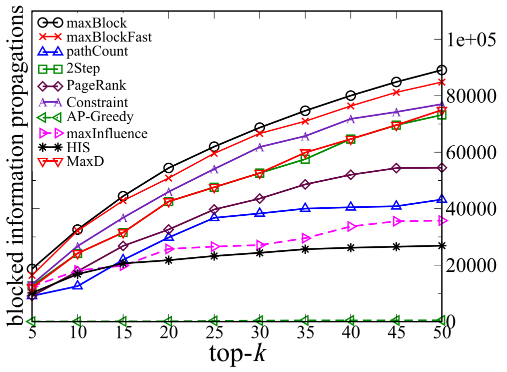
Fig. 11. Performance of the algorithms in the LiveJournal network.

hand, Fig. 10(b) shows that the running times of maxBlock and maxBlockFast with the network are approximately 7.5 and 0.5 h, respectively.

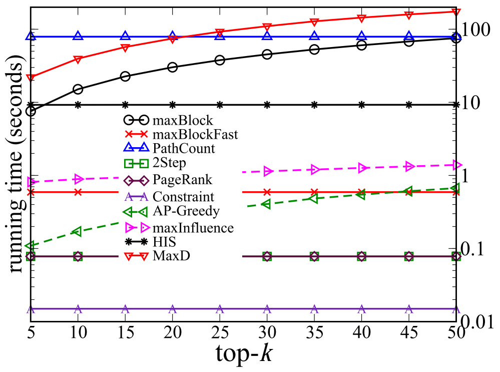
We finally studied the performance of the different algorithms in a large-scale network, i.e., LiveJournal, which consists of over 5 million nodes and 54 million edges. Fig. 11(a) shows that the top-k hole spanners found by maxBlock and maxBlockFast block at least 12% and 8% more information propagations than that by the best existing algorithm 2Step, respectively. Fig. 11(b) shows that it took approximately 7.4 and 0.45 days for maxBlock and maxBlockFast to deliver the solutions, respectively, when $k = 50$.

6.3. Algorithm performance with available ground-truth communities

Lou and Tang [23] proposed two algorithms HIS and MaxD for the top-k structural hole spanner problem, assuming that communities in a social network are given in advance. Given a social network Coauthor and its ground-truth communities, we here compare the performance of the proposed algorithms maxBlock and maxBlockFast against PathCount, 2Step, PageRank, Constraint, AP-Greedy, maxInfluence, HIS, and MaxD. Fig. 12(a) shows that the number of blocked information propagations by maxBlock is about 230% ($\approx \frac{89,090-26,900}{26,900}$) and 19% ($\approx \frac{89,090-74,900}{74,900}$) more than those by algorithms HIS and MaxD, respectively, when $k = 50$, while Fig. 12(b) shows that the running times of maxBlock and maxBlockFast are around 75 and 0.6 s, respectively, while the running times of algorithms HIS and MaxD are around 10 and 170 s, respectively.



(a) Blocked information propagations



(b) Running time

Fig. 12. Algorithm Performance in the `Coauthor` network.

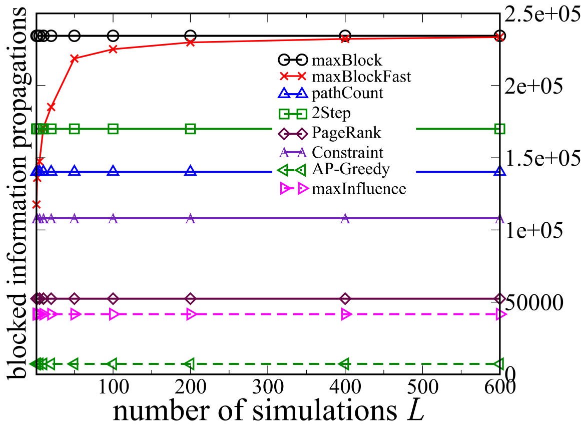


Fig. 13. Number of blocked information propagations by `maxBlockFast` with the growth of the number of simulations L in the `Twitter` network.

6.4. *The convergence of the heuristic algorithm*

The rest is to study the convergence of algorithm `maxBlockFast` in the `Twitter` social network, by increasing L from 1 to 500. Fig. 13 demonstrates that the number of blocked information propagations by algorithm `maxBlockFast` increases very fast by varying L from 1 to 50, and the number only slightly increases with more simulations. For example, the number of blocked information propagations by `maxBlockFast` is only about 50% of that by algorithm `maxBlock` when $L = 1$, but is as high as 93% of that by `maxBlock` when $L = 50$. This indicates that algorithm `maxBlockFast` converges very fast with the increase of L.

6.5. *Discussion*

The experimental results from Figs. 6 to 12 show that the removal of the structural hole spanners found by the proposed approximation algorithm `maxBlock` will block much more information propagations (up to 24%) than those by any other mentioned existing algorithms, which indicates that the hole spanners found by `maxBlock` not only bridge multiple communities but also maintain strong ties with their bridged communities. Moreover, the hole spanners found by the heuristic algorithm `maxBlockFast` blocked at least 92% information propagations compared with that by `maxBlock`.

The high quality of the hole spanners delivered by `maxBlock` and `maxBlockFast` was achieved at the expense of longer running time. For example, `maxBlock` took a longer running time than those of the other existing algorithms, which indicates that it is applicable for small and medium social networks. Moreover, algorithm `maxBlockFast` was around 13–300 times faster than `maxBlock`. In addition, although the running times of algorithm `maxBlockFast` were not short for large-scale social networks, e.g., approximately 0.5 h for the `DBLP-11` network with about 1 million nodes, and 0.45 days for the `LiveJournal` network with over 5 million nodes, it is noted that the algorithm ran on a single core of the server in our experiments. It is expected that it will run much faster if it runs on a multi-core server or a cluster of servers by performing multiple Monte-Carlo simulations simultaneously. In our future work, we will focus on further reducing the running times of the proposed algorithms.

7. Related work

With the exponential growth of various large-scale networks in the past decade (e.g., social networks, collaboration networks, and biological networks), the identification of important structural hole spanners from these networks has many potential applications, and numerous studies have been conducted in the past years [14,16,19,23,27,29,34,40].

Goyal et al. [14] formulated a structural hole spanner as a node that lies on many shortest paths among different pairs of nodes, similar to the betweenness centrality. However, it is time-consuming to find all-pairs shortest paths in a large network. As a result, Tang et al. [29] proposed to count the number of shortest paths with only a length of just two on which a node lies. Lou and Tang [23] introduced the very first model to find hole spanners in a social network, assuming that all communities in the network are given in advance. They argued that the removal of important structural hole spanners can significantly decrease the minimum cut of the communities, where such cut is the minimum number of edges such that their removals separate each community from its linked communities One major drawback of their model is that the identification of communities is very time-consuming and the communities are usually not known in advance, and thus, the quality of their delivered solution relies on the quality of the communities given. Rezvani et al. [27] observed that the removal of a node bridging multiple communities can significantly increase the communication cost in a network, which is the sum of all-pairs shortest distances in it. Xu et al. [34] later proposed fast, scalable algorithms for identifying important hole spanners, based on the work in [27]. He et al. [16] devised a harmonic modularity method to jointly detect communities and structural hole spanners, assuming that each user belongs to a single community only. This assumption, however, is not realistic since a user is usually in multiple communities in ground-truth social networks [35]. In addition, Zhang et al. [40] noted that two users may trust or distrust each other, and they found trust hole spanners. Chang et al. [8] studied the top-k structural diversity problem, which aims to find k nodes in a graph so that the sum of the structural diversities of the k nodes is maximized, where the structural diversity of a node is the number of connected components in the graph induced by the neighbors of the node.

It can be seen that most of these mentioned studies identified spanners by exploiting the topological characteristics of structural holes, e.g., whether a node can bridge multiple communities, the number of the communities, and the sizes of the communities. Burt [4,7], however, recently showed that the strength of the ties that connect a hole spanner to its bridged communities are highly correlated with the brokerage profit to the node. Unlike the aforementioned studies, in this study, we found structural hole spanners that not only bridge multiple communities but also have strong ties with their bridged communities. The found hole spanners thus play more important roles in real applications, particularly in blocking information propagations within social networks.

We also note that there are some studies on the competitive influence maximization problem [1,15], where the problem assumes that a negative opinion is already propagating in a social network, and we need to find k persons and propagate a positive opinion from them, such that the reduction in the number of negatively activated users is maximized. The problem, however, is totally different from the information propagation maximization problem in this paper, since it considers the information blocking of the negative opinion from only a portion of users, while we studied the problem of blocking the information propagation from all users.

8. Conclusions

In this paper, we studied the problem of identifying top-k structural hole spanners in a large-scale social network that not only bridge multiple communities but also have strong ties with their bridged communities. We first proposed an approximation algorithm for it, which delivers a $(1-\epsilon)$-approximate solution with a high probability $1-n^{-c}$, in an expected running time of $O(\frac{\epsilon^{-2}(k+c)k^2n^2m\log n}{OPT})$, where ϵ and c are two given constants with $0<\epsilon<1$ and $c\geq 1$, respectively. We also devised a fast, yet scalable, heuristic algorithm for the problem, which has a much shorter running time than that of the approximation algorithm, yet its solution is almost comparable with that of the latter. We finally evaluated the performance of the proposed algorithms through extensive experiments using real-world datasets. The experimental results showed that the proposed algorithms are very promising. Especially, the found hole spanners by the approximation algorithm blocked more than 24% information propagations compared to those by existing algorithms, and the spanners found by the heuristic algorithm blocked at least 92% information propagations compared with that by the approximation algorithm, while the running time of the former was only approximately $\frac{1}{300}$ to $\frac{1}{13}$ of the running time of the latter.

In our future work, we will focus on further reducing the running time of the proposed algorithms.

Declaration of interest

All authors have disclosed financial and personal relationships with other people and organizations in the manuscript file that could inappropriately influence (bias) their work.

Acknowledgments

The work by Wenzheng Xu was supported by the National Natural Science Foundation of China (grant nos. 61602330 and 61806134), the Sichuan Science and Technology Program (grant nos. 2018GZDZX0010, 2018GZ0094, 2018GZ0093, and 2017GZDZX0003), and the Fundamental Research Funds for the central Universities (grant no. 20822041B4104). The work by Weifa Liang was supported by the Australian Research Council Discovery Project (no. DP120102627). Moreover, the work by Jeffrey X. Yu was supported by the Research Grants Council of the Hong Kong SAR, China (no. 14221716).

Appendix A. Proof of NP-hardness

Lemma 12. *The top-k structural hole spanner problem is NP-hard.*

Proof. We show that the problem is NP-hard by a reduction from the NP-hard problem of calculating the information diffusion probability between a pair of nodes [18].

Given a directed graph $G = (V, E, w : E \mapsto [0, 1])$, consider the problem of computing the information diffusion probability $p_{u,\,v}$ from a node u to another node v in G. We reduce the problem to the top-k structural hole spanner problem as follows. We construct an auxiliary graph $G' = (V \cup \{v'\}, E', w' : E' \mapsto [0, 1])$ from G by adding a virtual node v' of node v, adding a directed edge (v, v') from v to v', and setting the activation probability $w'(v, v')$ to 1, i.e., $E' = E \cup \{(v, v')\}$ and $w'(v, v') = 1$. Assume that the information sharing frequency f_i of every node v_i is one, i.e., $f_i = 1$. It can be seen that the information diffusion probability $p'_{u,v'}$ from u to v' in G' is equal to $p_{u,\,v}$, and the removal of node v from G' will disconnect u from v'. Then, the expected number of blocked information propagations by node v from u to v' in G' is equal to the information diffusion probability $p_{u,\,v}$ in G. □

Appendix B. Proof of Lemma 2

Proof. Given a social network $G = (V, E, f : V \mapsto \mathcal{R}^+, w : E \mapsto [0, 1])$, recall that, within each of the L simulations, we first choose a starting node $s \in V$ randomly, and then generate a graph $G' = (V, E')$ from G under the live-edge graph model. The removal of a node u from G' results in n_u nodes becoming unreachable from s. To show that $\overline{H_L(u)}(= \frac{\sum_{l=1}^{L} X_l^u}{L} n)$ is an unbiased estimator, we only need to prove that, for each node $v \neq s$, the probability that the removal of node u from G blocks the information propagation from s to v under the independent cascade model, is equal to the probability that v is reachable from s in G' but becomes unreachable in $G' \backslash \{u\}$.

For node s and each node v in G with $s \neq v$, recall that $p_{s,\,v}$ and $p'_{s,v}$ are the probabilities that a piece of information successfully diffuses from node s to node v in graphs G and $G \backslash \{u\}$, respectively. Let $\Delta_{s,v} = p_{s,v} - p'_{s,v}$. Then, $\Delta_{s,\,v}$ is the probability that the removal of node u from G blocks the information propagation from s to v under the independent cascade model. We show that $\Delta_{s,\,v}$ is equal to the probability that v is reachable from s in G' but becomes unreachable in $G' \backslash \{u\}$ as follows.

For the original graph $G = (V, E)$, there are 2^m different subgraphs $G'_1, G'_2, \ldots, G'_{2^m}$ of G with the identical node set V, where $m = |E|$ is the number of edges in G. In each of the L simulations, a randomly generated graph G' under the live-edge graph model must be one of the 2^m subgraphs. Denote by $Pr[G' = G'_l]$ the probability that graph G' is equal to the lth subgraph G'_l among the 2^m subgraphs with $1 \leq l \leq 2^m$. Moreover, we use $I_{s,v}^{G'_l}$ to indicate whether node v is reachable from s in subgraph G'_l, i.e., $I_{s,v}^{G'_l} = 1$ if v is reachable from s; otherwise, $I_{s,v}^{G'_l} = 0$. Following Lemma 1, the information propagation probabilities $p_{s,\,v}$ in G and $p'_{s,v}$ in $G \backslash \{u\}$ are $p_{s,v} = \sum_{l=1}^{l=2^m} Pr[G' = G'_l] \cdot I_{s,v}^{G'_l}$ and $p'_{s,v} = \sum_{l=1}^{l=2^m} Pr[G' = G'_l] \cdot I_{s,v}^{G'_l \backslash \{u\}}$, respectively. Therefore,

$$\Delta_{s,v} = \sum_{l=1}^{l=2^m} Pr\left[G' = G'_l\right] \cdot \left(I_{s,v}^{G'_l} - I_{s,v}^{G'_l \backslash \{u\}}\right). \tag{B.1}$$

We calculate the value of $I_{s,v}^{G'_l} - I_{s,v}^{G'_l \backslash \{u\}}$ by dividing it into three cases. Case (1): Node v is unreachable from s in G'_l, i.e., $I_{s,v}^{G'_l} = 0$. Then, v is still unreachable from s in $G'_l \setminus \{u\}$, i.e., $I_{s,v}^{G'_l \backslash \{u\}} = 0$. We have that $I_{s,v}^{G'_l} - I_{s,v}^{G'_l \backslash \{u\}} = 0 - 0 = 0$. Case (2): Node v is reachable from s in both graphs G'_l and $G'_l \setminus \{u\}$. Then, we know that $I_{s,v}^{G'_l} - I_{s,v}^{G'_l \backslash \{u\}} = 1 - 1 = 0$. Case (3): Node v is reachable from s in G'_l but is unreachable from s in $G'_l \setminus \{u\}$ owing to the removal of node u from G'_l. Then, $I_{s,v}^{G'_l} - I_{s,v}^{G'_l \backslash \{u\}} = 1 - 0 = 1$.

It can be seen that only in Case (3), the value of $I_{s,v}^{G'_l} - I_{s,v}^{G'_l \backslash \{u\}}$ is non-zero, which means that $\Delta_{s,\,v}$ is equal to the probability that v is reachable from s in G' but becomes unreachable in $G' \backslash \{u\}$. The lemma then follows. □

Appendix C. Proof of Lemma 5

Proof. We only show the upper bound on $Pr[\overline{H_L(S)} - H(S) \geq \epsilon \cdot H(S)]$, since the analysis of the upper bound on $Pr[\overline{H_L(S)} - H(S) \leq -\epsilon \cdot H(S)]$ can be done similarly and was thus omitted.

Notice that, although the L random variables $\overline{X_l^{u_i}}, \overline{X_2^{u_i}}, \ldots, \overline{X_L^{u_i}}$ are independent of each other, the k random variables $\overline{H_L(u_1)}, \overline{H_L(u_2)}, \ldots, \overline{H_L(u_k)}$ are dependent on, rather than independent of each other.

For any node $u_i \in S$, we first analyze the upper bound on $Pr[\overline{H_L(u_i)} - H(u_i) \geq \epsilon \cdot \frac{H(S)}{k}]$.

$$\begin{aligned}
&Pr\left[\overline{H_L(u_i)} - H(u_i) \geq \epsilon \cdot \frac{H(S)}{k}\right] \\
&= Pr\left[\frac{\overline{H_L(u_i)}}{n(n-1)} - a_i \geq \frac{\epsilon \cdot a_S}{k}\right], \text{ as } a_i = \frac{H(u_i)}{n(n-1)}, a_S = \frac{H(S)}{n(n-1)} \\
&= Pr\left[\frac{\overline{H_L(u_i)}}{n(n-1)} - a_i \geq \frac{\epsilon \cdot a_S}{ka_i} a_i\right] \\
&\leq exp\left(-\frac{(\frac{\epsilon \cdot a_S}{ka_i})^2}{2 + \frac{\epsilon \cdot a_S}{ka_i}} La_i\right), \text{ by Chernoff–Hoeffding bounds, where } \delta = \frac{\epsilon \cdot a_S}{ka_i} \\
&= exp\left(-\frac{\epsilon^2 a_S^2}{2k^2 a_i + \epsilon k a_S} L\right), \\
&\leq exp\left(-\frac{\epsilon^2 a_S}{(2k+\epsilon)k} L\right), \text{ as } a_i \leq a_S = \sum_{j=1}^{k} a_j \\
&\leq exp\left(-\frac{\epsilon^2 L}{(2k+\epsilon)kn^2} H(S)\right), \text{ as } a_S = \frac{H(S)}{n(n-1)} \geq \frac{H(S)}{n^2}.
\end{aligned} \tag{C.1}$$

We then show the upper bound on $Pr[\overline{H_L(S)} - H(S) \geq \epsilon \cdot H(S)]$.

$$\begin{aligned}
&Pr\left[\overline{H_L(S)} - H(S) \geq \epsilon \cdot H(S)\right] \\
&= Pr\left[\sum_{i=1}^{k} \overline{H_L(u_i)} - \sum_{i=1}^{k} H(u_i) \geq \epsilon \cdot H(S)\right] \\
&\leq Pr\left[\overline{H_L(u_1)} - H(u_1) \geq \epsilon \cdot \frac{H(S)}{k}, \text{ or } \overline{H_L(u_2)} - H(u_2) \geq \epsilon \cdot \frac{H(S)}{k}, or, \ldots, \text{ or } \overline{H_L(u_k)} - H(u_k) \geq \epsilon \cdot \frac{H(S)}{k}\right] \\
&\leq \sum_{i=1}^{k} Pr\left[\overline{H_L(u_i)} - H(u_i) \geq \epsilon \cdot \frac{H(S)}{k}\right], \text{ by the union bound} \\
&\leq k \cdot exp\left(-\frac{\epsilon^2 L}{(2k+\epsilon)kn^2} H(S)\right), \text{ by Inequality (C.1).}
\end{aligned} \tag{C.2}$$

Inequality (8) then follows. On the other hand, Inequality (9) can be shown similarly and was thus omitted. The lemma then follows. □

Appendix D. Proof of Lemma 6

Proof.

$$\begin{aligned}
&Pr\left[|\overline{H_L(S)} - H(S)| \geq \frac{\epsilon}{2} OPT\right] \\
&= Pr\left[|\overline{H_L(S)} - H(S)| \geq \frac{\epsilon \cdot OPT}{2H(S)} H(S)\right] \\
&\leq 2k \cdot exp\left(-\frac{(\frac{\epsilon \cdot OPT}{2H(S)})^2 L}{(2k + \frac{\epsilon \cdot OPT}{2H(S)})kn^2} H(S)\right), \text{ by Lemma 5} \\
&= 2k \cdot exp\left(-\frac{\epsilon^2 L}{(8k \cdot H(S) + 2\epsilon \cdot OPT)kn^2} OPT^2\right)
\end{aligned}$$

$$\begin{aligned}
&\le 2k \cdot exp\left(-\frac{\epsilon^2 L}{(8k+2\epsilon)kn^2} OPT\right), \text{ as } H(S) \le OPT \\
&\le kn^{-(k+c)}, \text{ as } L \ge \frac{((k+c)\log n + \log 2)(8k+2\epsilon)kn^2}{\epsilon^2 \cdot OPT} \\
&\le n^{-c} / \binom{n}{k}, \text{ as } k\binom{n}{k} \le n^k.
\end{aligned} \tag{D.1}$$

The lemma then follows. □

Appendix E. Proof of Lemma 7

Proof. Following Lemma 6, the probability $Pr[|\overline{H_L(S)} - H(S)| \le \frac{\epsilon}{2} OPT]$ for all subsets with k nodes in V is no less than $1 - n^{-c}$ by the union bound, since there are $\binom{n}{k}$ sets with k nodes. Considering the node set S found by Algorithm 3 and the optimal solution S^*, we have

$$\begin{aligned}
H(S) &\ge \overline{H_L(S)} - \frac{\epsilon}{2}\ OPT \\
&\ge \overline{H_L(S^*)} - \frac{\epsilon}{2}\ OPT, \text{ as } \overline{H_L(S)} \ge \overline{H_L(S^*)} \text{ by Algorithm 1} \\
&\ge H(S^*) - \frac{\epsilon}{2}\ OPT - \frac{\epsilon}{2}\ OPT \\
&= (1-\epsilon)\ OPT, \text{ as } H(S^*) = OPT.
\end{aligned}$$

□

Appendix F. Proof of Lemma 8

Proof.

$$\begin{aligned}
& Pr\left[\overline{H_{L_t}(S)} \ge OPT_g\right] \\
&= Pr\left[\overline{H_{L_t}(S)} - H(S) \ge OPT_g - H(S)\right] \\
&\le Pr\left[\overline{H_{L_t}(S)} - H(S) \ge OPT_g - OPT\right], \\
&\quad \text{by Observation 1}, as\ OPT_g - H(S) \ge OPT_g - OPT \\
&= Pr\left[\overline{H_{L_t}(S)} - H(S) \ge \frac{OPT_g - OPT}{OPT}\ OPT\right] \\
&\le Pr\left[\overline{H_{L_t}(S)} - H(S) \ge 2^{s-t-1}\ OPT\right], \\
&\quad \text{as } \frac{OPT_g}{OPT} - 1 \ge \frac{\frac{OPT_{ub}}{2^t}}{\frac{OPT_{ub}}{2^s}} - 1 = 2^{s-t} - 1 \ge 2^{s-t-1}, \\
&\quad OPT_g = \frac{OPT_{ub}}{2^t}, OPT \le \frac{OPT_{ub}}{2^s}, \text{ and } t \le s-1 \\
&\le k \cdot exp\left(-\frac{\epsilon^2 L_t}{(2k+\epsilon)kn^2}\ OPT\right), \text{ by Lemma 6, where} \epsilon = 2^{s-t-1} \\
&\le k \cdot exp\left(-\frac{\epsilon L_t}{(2k+1)kn^2}\ OPT\right), \\
&\quad \text{as} \frac{2k}{\epsilon} \le 2k, \epsilon = 2^{s-t-1} \ge 1, t \le s-1 \\
&= k \cdot exp\left(-\frac{2^{s-t-1}\frac{\lambda}{OPT_g}}{(2k+1)kn^2}\ OPT\right), \text{ as } L_t = \frac{\lambda}{OPT_g} \\
&\le k \cdot exp\left(-\frac{\lambda}{4(2k+1)kn^2}\right), \text{ as } \frac{OPT}{OPT_g} \ge \frac{\frac{OPT_{ub}}{2^{s+1}}}{\frac{OPT_{ub}}{2^t}} = \frac{2^t}{2^{s+1}} \\
&= \frac{n^{-c}}{T}, \text{ as } \lambda = 4(c\ \log n + \log kT)(2k+1)kn^2.
\end{aligned} \tag{F.1}$$

The lemma then follows. □

Appendix G. Proof of Lemma 9

Proof.

$$
\begin{aligned}
& Pr\left[\overline{H_{L_t}(S)} < OPT_g\right] \\
\leq\ & Pr\left[\overline{H_{L_t}(S^*)} < OPT_g\right], \text{ by Observation 2} \\
=\ & Pr\left[\overline{H_{L_t}(S^*)} - OPT < OPT_g - OPT\right] \\
=\ & Pr\left[\overline{H_{L_t}(S^*)} - H(S^*) < \frac{OPT_g - OPT}{OPT} H(S^*)\right] \\
\leq\ & Pr\left[\overline{H_{L_t}(S^*)} - H(S^*) < -\frac{1}{2} H(S^*)\right], \\
& \text{as } \frac{OPT_g}{OPT} - 1 \leq \frac{OPT_{ub}/2^t}{OPT_{ub}/2^{s+1}} - 1 = \frac{2^{s+1}}{2^t} - 1 \leq \frac{2^{s+1}}{2^{s+2}} - 1 = -\frac{1}{2}, t \geq s+2 \\
\leq\ & k \cdot exp\left(-\frac{L_t}{8k^2n^2}\ OPT\right), \text{ by Lemma 5, where } \epsilon = \frac{1}{2} \\
\leq\ & k \cdot exp\left(-\frac{\frac{\lambda}{OPT_g}}{8k^2n^2}\ OPT\right), \text{ as } L_t = \frac{\lambda}{OPT_g} \\
\leq\ & k \cdot exp\left(-\frac{2^{t-s-1}\lambda}{8k^2n^2}\right), \text{ as } \frac{OPT}{OPT_g} \geq \frac{\frac{OPT_{ub}}{2^{s+1}}}{\frac{OPT_{ub}}{2^t}} = 2^{t-s-1} \\
=\ & k \cdot exp\left(-2^{t-s-1}(c\log n + \log kT)(1 + \frac{1}{2k})\right), \\
& \text{as } \lambda = 4(c\log n + \log kT)(2k+1)kn^2 \\
\leq\ & k \cdot exp\left(-2^{t-s-1}(c\ \log n + \log kT)\right) \\
=\ & kn^{-c\cdot 2^{t-s-1}}(kT)^{-2^{t-s-1}} \\
\leq\ & kn^{-c\cdot 2^{t-s-1}}(kT)^{-1}, \text{ as } 2^{t-s-1} \geq 1, t \geq s+2 \\
=\ & \frac{n^{-c\cdot 2^{t-s-1}}}{T} \\
\leq\ & \frac{n^{-c}}{2^{t-s-1}T}, \text{ as } (n^c)^{2^{t-s-1}} \geq n^c 2^{t-s-1}, t \geq s+2.
\end{aligned}
\tag{G.1}
$$

The lemma then follows. □

Appendix H. Proof of Lemma 10

Proof. Algorithm 2 proceeds iteratively. It may terminate at the 1st, 2nd, …, or Tth iteration. On one hand, when $1 \leq t \leq s-1$, the probability that Algorithm 2 stops at the tth iteration is

$$
\begin{aligned}
& Pr\left[\overline{H_{L_1}(S)} < \frac{OPT_{ub}}{2^1}, \overline{H_{L_2}(S)} < \frac{OPT_{ub}}{2^2}, \ldots, \overline{H_{L_{t-1}}(S)} < \frac{OPT_{ub}}{2^{t-1}}, \overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}\right] \\
\leq\ & Pr\left[\overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}\right] \\
\leq\ & \frac{n^{-c}}{T}, \text{ by Inequality (F.1).}
\end{aligned}
\tag{H.1}
$$

On the other hand, when $s+3 \leq t \leq T$, the probability that Algorithm 2 stops at the tth iteration is

$$
\begin{aligned}
& Pr\left[\overline{H_{L_1}(S)} < \frac{OPT_{ub}}{2^1}, \overline{H_{L_2}(S)} < \frac{OPT_{ub}}{2^2}, \ldots, \overline{H_{L_{t-1}}(S)} < \frac{OPT_{ub}}{2^{t-1}}, \overline{H_{L_t}(S)} \geq \frac{OPT_{ub}}{2^t}\right] \\
\leq\ & Pr\left[\overline{H_{L_{t-1}}(S)} < \frac{OPT_{ub}}{2^{t-1}}\right] \\
\leq\ & \frac{n^{-c}}{2^{t-s-2}T}, \text{ by Inequality (G.1), as } t-1 \geq s+2.
\end{aligned}
\tag{H.2}
$$

The probability that Algorithm 2 does not stop at the sth, $(s+1)$th, or $(s+2)$th iteration is no more than $\sum_{t=1}^{s-1} \frac{n^{-c}}{T} + \sum_{t=s+3}^{T} \frac{n^{-c}}{2^{t-s-2}T} \leq n^{-c}$ by the union bound in Lemma 4. In other words, the probability that Algorithm 2 stops at the sth,

$(s+1)$th, or $(s+2)$th iteration is no less than $1-n^{-c}$. In this case, $OPT' \geq \frac{OPT_{ub}}{2^{s+2}}/2 = \frac{1}{8}\frac{OPT_{ub}}{2^s} \geq \frac{OPT}{8}$, as $OPT \leq \frac{OPT_{ub}}{2^s}$. Moreover, $OPT' \leq \frac{OPT_{ub}}{2^s}/2 = \frac{OPT_{ub}}{2^{s+1}} \leq OPT$. Then, $\frac{OPT}{8} \leq OPT' \leq OPT$.

We finally analyze the time complexity of Algorithm 2. Assume that it stops at the tth iteration with $1 \leq t \leq T$. Then, the number of simulations performed is $\sum_{t'=1}^{t} L_{t'} = \sum_{t'=1}^{t} \frac{\lambda}{\frac{OPT_{ub}}{2^{t'}}} = O(\frac{\lambda}{OPT_{ub}}2^{t+1})$. The expected number of simulations performed by the algorithm is thus no more than

$$\begin{aligned}
&\sum_{t=1}^{s-1}\frac{n^{-c}}{T}\cdot O\left(\frac{\lambda}{OPT_{ub}}2^{t+1}\right)+\sum_{t=s}^{s+2}1\cdot O\left(\frac{\lambda}{OPT_{ub}}2^{t+1}\right)+\sum_{t=s+3}^{T}\frac{n^{-c}}{2^{t-s-2}T}\cdot O\left(\frac{\lambda}{OPT_{ub}}2^{t+1}\right)\\
&\leq O\left(\frac{n^{-c}}{T}\frac{\lambda}{OPT_{ub}}2^s\right)+O\left(\frac{\lambda}{OPT_{ub}}2^s\right)+\sum_{t=s+3}^{T}O\left(\frac{n^{-c}}{T}\frac{\lambda}{OPT_{ub}}2^s\right)\\
&=O\left(\frac{\lambda}{OPT_{ub}}2^s\right)\\
&\leq O\left(\frac{\lambda}{OPT}\right),\ \text{as } OPT \leq \frac{OPT_{ub}}{2^s}\\
&=O\left(\frac{(c\log n+\log kT)k^2n^2}{OPT}\right),\ \text{as } \lambda = 4(c\log n+\log kT)(2k+1)kn^2\\
&=O\left(\frac{k^2n^2(\log k+c\log n)}{OPT}\right),\ \text{as } T=\left\lceil \log_2\frac{OPT_{ub}}{\Delta/2}\right\rceil = O(\log(kn^2)).
\end{aligned} \tag{H.3}$$

On the other hand, in each simulation, it takes $O(m)$ time to find the immediate dominator of each node in an s-rooted directed graph, by invoking an algorithm in [11]. Therefore, the expected time of Algorithm 2 is $O(\frac{k^2n^2(\log k+c\log n)}{OPT})\cdot O(m) = O(\frac{k^2n^2m(\log k+c\log n)}{OPT})$. □

Appendix I. Proof of Lemma 11

Proof. On one hand, assume that a node u_s is a separation node in G_s. It can be easily seen that the aforementioned three conditions are met.

On the other hand, assume that a node u_s meets the three conditions. However, u_s is not a separation node. Then, there must be an ancestor u_A^s and a descendant u_D^s of u^s in G_s such that there is a directed simple path P in G_s from u_A^s to u_D^s, and path P does not contain node u^s. Moreover, since u_A^s is an ancestor of u^s, there is a directed simple path P_1 in G_s from u_A^s to u^s, assuming that $P_1 = u_A^s \to \cdots \to v_1^s \to u^s$. Similarly, there is a directed simple path P_2 in G_s from u^s to u_D^s, assuming that $P_2 = u^s \to v_2^s \to \cdots \to u_D^s$, where v_1^s is an incoming neighbor of u^s and v_2^s is an outgoing neighbor of u^s. Consider the underlying undirected graph G_s' of G_s. There is a path between nodes v_1^s and v_2^s detouring node u^s in G_s', e.g., $v_1^s \to \cdots \to u_A^s \to P \to u_D^s \to \cdots \to v_2^s$. Then, after the removal of node u^s from G_s', nodes v_1^s and v_2^s are still in the same connected component in $G_s' \setminus \{u^s\}$, and thus, they are in the same group. This contradicts the assumption that all neighbors of u^s in the same group are either incoming neighbors or outgoing neighbors exclusively. Therefore, the supposition that u_s is not a separation node is incorrect. The lemma then follows. □

References

[1] A. Bozorgi, S. Samet, J. Kwisthout, T. Wareham, Community-based influence maximization in social networks under a competitive linear threshold model, Knowl.-Based Syst. 134 (2017) 149–158.
[2] C. Budak, D. Agrawal, A.E. Abbadi, Limiting the spread of misinformation in social networks, in: Proc. ACM Int. Conf. World Wide Web (WWW), 2011, pp. 665–674.
[3] R.S. Burt, Structural Holes: The Social Structure of Competition, Harvard University Press, 1995.
[4] R.S. Burt, Structural holes versus network closure as social capital, Soc. Cap. (2001).
[5] R.S. Burt, Structural holes and good ideas, Am. J. Sociol. 110 (2) (2004) 349–399.
[6] R.S. Burt, Secondhand brokerage: evidence on the importance of local structure for managers, bankers, and analysts, Acad. Manag. J. 50 (1) (2007) 119–148.
[7] R.S. Burt, Structural Holes in Virtual Worlds, Oxford University Press, 2017.
[8] L. Chang, C. Zhang, X. Lin, L. Qin, Scalable top-k structural diversity search, in: Proc. IEEE 33rd Int. Conf. Data Eng. (ICDE), 2017, pp. 95–98.
[9] W. Chen, L.V.S. Lakshmanan, C. Castillo, Information and Influence Propagation in Social Networks, Morgan & Claypool Publishers, 2013.
[10] J. Chen, B. Yuan, Detecting functional modules in the yeast protein-protein interaction network, Bioinformatics 22 (18) (2006) 2283–2290.
[11] L. Georgiadis, R.E. Tarjan, Finding dominators revisited: extended abstract, in: Proc. 15th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA), 2004, pp. 869–878.
[12] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Nat. Acad. Sci. 99 (12) (2002) 7821–7826.
[13] K.I. Goh, M.E. Cusick, D. Valle, B. Childs, M. Vidal, A.L. Barabási, The human disease network, Proc. Nat. Acad. Sci. 104 (21) (2007) 8685–8690.
[14] S. Goyal, F. Vega-Redondo, Structural holes in social networks, J. Econ. Theory 137 (1) (2007) 460–492.
[15] X. He, G. Song, W. Chen, Q. Jiang, Influence blocking maximization in social networks under the competitive linear threshold model, in: Proc. SIAM Int. Conf. Data Mining, 2012, pp. 463–474.

[16] L. He, C. Lu, J. Ma, J. Cao, L. Shen, P.S. Yu, Joint community and structural hole spanner detection via harmonic modularity, in: Proc. 22nd ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD), 2016, pp. 875–884.
[17] E. Katz, The two-step flow of communication: an up-to-date report on an hypothesis, Public Opin. Q. 21 (1) (1957) 61–78.
[18] D. Kempe, J. Kleinberg, E. Tardos, Maximizing the spread of influence through a social network, in: Proc. 9th ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD), 2003, pp. 137–146.
[19] J. Kleinberg, S. Suri, E. Tardos, T. Wexler, Strategic network formation with structural holes, in: Proc. 9th ACM Conf. Electron. Commerce, 2008, pp. 284–293.
[20] Y.Y. Ko, K.J. Cho, S.W. Kim, Efficient and effective influence maximization in social networks: a hybrid-approach, Inf. Sci. 465 (2018) 144–161.
[21] S. Kumar, J. Cheng, J. Leskovec, Antisocial behavior on the web: characterization and detection, in: Proc. 26th ACM Int. Conf. World Wide Web (WWW) Companion, 2017, pp. 947–950.
[22] J. Li, X. Wang, K. Deng, X. Yang, T. Sellis, J.X. Yu, Most influential community search over large social networks, in: Proc. IEEE 33rd Int. Conf. Data Eng. (ICDE), 2017, pp. 871–882.
[23] T. Lou, J. Tang, Mining structural hole spanners through information diffusion in social networks, in: Proc. ACM Int. Conf. World Wide Web (WWW), 2013, pp. 825–836.
[24] M.V. Marathe, A.K.S. Vullikanti, Computational epidemiology, Commun. ACM 56 (7) (2013) 88–96.
[25] M. Mitzenmacher, E. Upfal, Probability and Computing: Randomized Algorithms and Probabilistic Analysis, Cambridge University Press, 2005.
[26] L. Page, S. Brin, R. Motwani, T. Winograd, The pagerank citation ranking: bringing order to the web, Tech. Rep. Stanford InfoLab (1999).
[27] M. Rezvani, W. Liang, W. Xu, C. Liu, Identifying top-k structural hole spanners in large-scale social networks, in: Proc. 24th ACM Int. Conf. Inform. Knowl. Manag. (CIKM), 2015, pp. 263–272.
[28] Y. Sun, C. Qian, N. Yang, P.S. Yu, Collaborative inference of coexisting information diffusions, in: Proc. IEEE Int. Conf. Data Mining (ICDM), 2017, pp. 1093–1098.
[29] J. Tang, T. Lou, J. Kleinberg, Inferring social ties across heterogeneous networks, in: Proc. ACM Int. Conf. Web Search Data Mining (WSDM), 2012, pp. 743–752.
[30] Y. Tang, Y. Shi, X. Xiao, Influence maximization in near-linear time: a martingale approach, in: Proc. ACM Int. Conf. Manag. Data (SIGMOD), 2015, pp. 1539–1554.
[31] H. Tong, S. Papadimitriou, C. Faloutsos, P.S. Yu, T. Eliassi-Rad, Gateway finder in large graphs: problem definitions and fast solutions, Inf. Retr, 15 (3–4) (2012) 391–411.
[32] S. Vosoughi, D. Roy, S. Aral, The spread of true and false news online, Science, 359 (2018) 1146–1151.
[33] W. Xu, W. Liang, X. Lin, J.X. Yu, Finding top-k influential users in social networks under the structural diversity model, Inf. Sci. 355–356 (2016) 110–126.
[34] W. Xu, M. Rezvani, W. Liang, J.X. Yu, C. Liu, Efficient algorithms for the identification of top-k structural hole spanners in large social networks, IEEE Trans. Knowl. Data Eng. 29 (5) (2017) 1017–1030.
[35] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowl. Inf. Syst. 42 (2015) 181–213.
[36] H. Zhang, M.A. Alim, X. Li, M.T. Thai, H.T. Nguyen, Misinformation in online social networks: detect them all with a limited budget, ACM Trans. Inf. Syst. 34 (3) (2016). Article 18.
[37] A. Zareie, A. Sheikhahmadi, M. Jalili, Identification of influential users in social networks based on users' interest, Inf. Sci. 493 (2019) 217–231.
[38] Y. Zhang, J.X. Yu, Unboundedness and efficiency of truss maintenance in evolving graphs, in: Proc. ACM SIGMOD/PODS Int. Conf. Manag. Data, 2019.
[39] Z. Zheng, F. Ye, R.H. Li, Finding weighted k-truss communities in large networks, Inf. Sci. 417 (2017) 344–360.
[40] J. Zhang, Q. Zhan, L. He, C.C. Aggarwal, P.S. Yu, Trust hole identification in signed networks, in: Proc. European Conf. Mach. Learning Principles Practice Knowl. Discovery (ECML-PKDD), 2016, pp. 697–713.

Wenzheng Xu received the B.Sc., ME, and Ph.D. degrees in computer science from Sun Yat-Sen University, Guangzhou, P.R. China, in 2008, 2010, and 2015, respectively. He cur- rently is an Associate Professor at Sichuan University and was a visitor at the Australian National University. His research interests include online social networks, wireless ad hoc and sensor networks, mobile computing, approximation algorithms, combinatorial optimiza- tion, and graph theory.

Tong Li received the B.Sc. degree in computer science from Sichuan University, P.R. China, in 2015. She currently is a third year mater student in computer science at Sichuan University. Her research interests include wireless sensor networks and social networks.

Weifa Liang received the Ph.D. degree from the Australian National University in 1998, the ME degree from the University of Science and Technology of China in 1989, and the B.Sc. degree from Wuhan University, China in 1984, all in computer science. He is currently a professor at the Australian National University. His research interests include wireless ad hoc and sensor networks, cloud computing, software-defined networking, online social net- works, design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory.

Jeffrey Xu Yu received the BE, ME, and Ph.D. degrees in computer science, from the University of Tsukuba, Tsukuba, Japan, in 1985, 1987, and 1990, respectively. He is cur- rently 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.

Ning Yang received his Ph.D degree in Computer Science from Sichuan University in 2010. He is an associate professor at Sichuan University and was a visitor at the University of illinois at Chicago. His research interests include social network analysis, recommender system, and heterogeneous information network analysis.

Shaobing Gao received his Ph.D. degree from UESTC, Chengdu, China, in 2017. He is currently an associate professor in College of Computer Science, Sichuan University. His research interests include biologically inspired vision and image processing, approximation algorithms, and combinatorial optimization. He has authored or co-authored over 15 articles on international high-impact journals and conferences including TPAMI, TIP, CVPR, ICCV, ECCV, and NIPS.