The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Forbidden Nodes Aware Community Search

Chaokun Wang,[*] **Junchao Zhu**
School of Software, Tsinghua University, Beijing 100084, China
chaokun@tsinghua.edu.cn, zhu-jc17@mails.tsinghua.edu.cn

Abstract

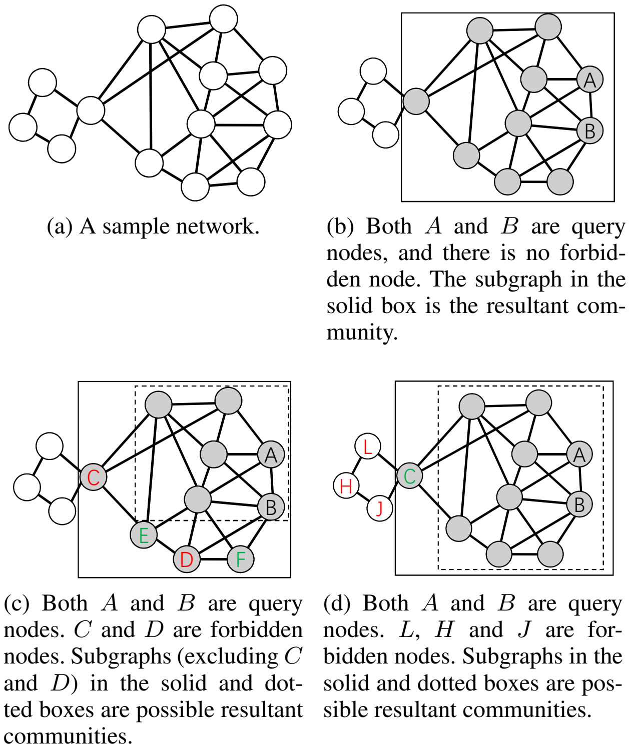
Community search is an important problem in network analysis, which has attracted much attention in recent years. It starts with some given nodes, pays more attention to local network structures, and gets personalized resultant communities quickly. In this paper, we argue that there are many real scenarios where some nodes are not allowed to appear in the community. Then, we introduce a new concept called forbidden nodes and present a new problem of forbidden nodes aware community search to describe these scenarios.

To address the above problem, three methods are proposed, i.e., k-core based FORTE (Forbidden nOdes awaRe communiTy sEarch), k-truss based FORTE and C_W based FORTE, where the effects of both forbidden nodes and query nodes are thoroughly considered for each node in the resultant community. The former two methods are able to make use of popular community structures, while the latter is based on a new metric called weighted conductance. The extensive experiments conducted on real data sets demonstrate the effectiveness of the proposed methods.

Motivation

Complex networks (e.g., social networks) never fail to fascinate human beings. As an important approach to giving insights into a complex network, the research on community structures attracts more and more attention these years. One of the important topics in this field is community search, a.k.a. local community detection, which aims to find out a cohesive subnetwork (i.e., subgraph) containing given nodes.

Community search enables users to get personalized local cohesive structures faster, especially in large scale networks. A large number of methods for community search have been proposed so far, such as topology-based methods and semantics-enhanced methods. Topology-based methods focus on the topological structure of communities in a network, such as k-core and k-truss. Semantics-enhanced methods consider both attributes of nodes or edges of the network and the topology of the network, and then define communities in a more semantic way.



(a) A sample network.

(b) Both A and B are query nodes, and there is no forbidden node. The subgraph in the solid box is the resultant community.

(c) Both A and B are query nodes. C and D are forbidden nodes. Subgraphs (excluding C and D) in the solid and dotted boxes are possible resultant communities.

(d) Both A and B are query nodes. L, H and J are forbidden nodes. Subgraphs in the solid and dotted boxes are possible resultant communities.

Figure 1: Examples for community search with forbidden nodes. Nodes in gray color form a cohesive subgraph.

However, to the best of our knowledge, all existing methods for community search just emphasize the occurrence of some nodes (called query nodes in this work), i.e., only nodes that should be included in the resultant community are considered as the input. Then, users are not able to express lot of real and appropriate needs, such as preventing some specific objects from appearing in the resultant communities. Therefore, in this study we argue that the nonexistence of some nodes (called forbidden nodes in this work) should also be considered as the input of the community search task.

Example 1. *A sample network is given in Figure 1(a). As shown in Figure 1(b), nodes A and B are treated as the query nodes of a community search task. Here, the cohesive-*

[*] Corresponding author: Chaokun Wang.

ness is measured by a 3-core structure, which means each node of a resultant community has at least three neighbors in this community. Then, the subgraph in the solid box is an answer since it is such a local structure containing nodes A and B.

As illustrated in Figure 1(c), nodes A and B are still query nodes while nodes C and D are forbidden nodes. Simply deleting C and D from the subgraph in the solid box (which is generated in Figure 1(b)) seems to be direct but not proper, since the remaining subgraph in the solid box contains nodes E and F whose degrees are lower than 3 after the deletion, i.e., it is not a 3-core anymore. The subgraph in the dotted box is more proper because the nodes inside the box still have three or more neighbors.

As described in Example 1, it is not a good idea to directly delete forbidden nodes from the result generated by the existing community search methods. In addition, we argue that the relation between each remaining node and forbidden nodes as well as that between it and query nodes should be thoroughly considered. For instance, each node inside the resultant community should be closer to query nodes than forbidden nodes.

Example 2. *As depicted in Figure 1(d), nodes A and B are query nodes; nodes L, H and J are forbidden nodes. It seems that the subgraph in the solid box is a good result since it does not contain any forbidden nodes. However, things may not be that simple in practice. Suppose it is a movie network where edges represent the co-watched relations, and nodes A and B have been watched by users before. Also, nodes in the solid box represent fiction movies, and forbidden nodes are horror movies. In this context, node C is likely to be a fiction movie with a lot of horrible elements. Therefore it would be better not to include node C in the final community especially in an online movie recommendation system, which means the subgraph in the dotted box is a better choice.*

From the above two examples, it is clear that simply excluding forbidden nodes from the communities to be generated cannot provide users with desirable resultant communities. The introduction of forbidden nodes inspires us to rethink whether the nodes in a network should appear in the resultant community, especially those that are closely related to the forbidden nodes. It is the biggest challenge of the present study.

To address this challenge, in this paper three methods are proposed to find communities containing query nodes without forbidden nodes. Especially, the effect of forbidden nodes is fully considered and better community results are generated. The main contributions of this paper are summarized as follows:

- A novel concept called forbidden nodes is introduced to community search problems so that users of complex networks are able to express more realistic needs. Meanwhile, the problem of forbidden nodes aware community search is presented.
- Three algorithms, i.e., k-core based FORTE (Forbidden nOdes awaRe communiTy sEarch), k-truss based FORTE and weighted-conductance (denoted as C_W in the rest of this paper) based FORTE, are proposed to find a community containing query nodes without forbidden nodes, where the effects of forbidden nodes as well as query nodes are well considered.
- The extensive experiments conducted on real data sets demonstrate the effectiveness of the proposed methods.

Related Work

Community search. Given a set of nodes, the task of community search seeks communities that contain them. Due to the focus on local community structure, community search can efficiently find the communities where the nodes that users care about are located. The classical community search algorithms are mainly based on specific structures like k-clique (Cui et al. 2013), k-core (Sozio and Gionis 2010; Cui et al. 2014), k-truss (Huang et al. 2014; Akbas and Zhao 2017), and densely connected subgraphs (Wu et al. 2015). For example, Cui et al. proposed the problem of looking for a community with minimal degree k containing a given node and the corresponding algorithm (Cui et al. 2014). Huang et al. proposed the community definition based on k-truss and designed the tcp-index to find the target community (Huang et al. 2014).

In addition, there are semantics-enhanced community search methods combining network topology and node attributes. For example, Shang et al. constructed a TA-graph based on the similarity of node attributes as well as that of node topology, and then proposed an effective community search algorithm AGAR (Shang et al. 2018). Fang et al. designed an index structure called CL-tree, on the basis of k-core, requiring the nodes in the community to share as many attributes as possible (Fang et al. 2016). Huang et al. designed a scoring function based on k-truss to measure the popularity of a given attribute in the community, and proposed the Attribute-Truss community definition (Huang and Lakshmanan 2017). Chen et al. considered the constraint of users' spatial information in k-truss search named co-located community search (Chen et al. 2018).

However, all these proposed community search methods ignore the existence of forbidden nodes which frequently appear in user requirements. At the same time, the influence of forbidden nodes has never been considered before.

Community detection. The community detection problem has been widely studied, which aims at finding out all communities in a given network. Typical methods for community detection mainly include partitioning, clustering, label propagation, and so on (Wang et al. 2015).

Partitioning methods directly decompose the original network into disconnected subgraphs, such as KMF algorithm (Zhao and Tung 2012) and SCD algorithm (Prat-Pérez, Dominguez-Sal, and Larriba-Pey 2014). KMF removes edges that take part in less than k triangles. SCD algorithm partitions the network by maximizing the weighted community clustering.

Clustering methods can be divided into hierarchy clustering, spectral clustering and k-means clustering. Hierarchy clustering firstly constructs a hierarchical tree and then cuts it at certain level to optimize the goodness of community structures. Fast-Newman (Newman 2004), CNM (Clauset, Newman, and Moore 2004), Radicchi (Radicchi et al. 2004), GN (Newman and Girvan 2004) are all typical hierarchy clustering methods. Spectral clustering is based on the eigenvectors of adjacent matrix (Shi and Malik 2000; Jin and others 2015). k-means clustering is a common clustering method. In the community detection problem, the similarity of two nodes is defined by shortest-path distance (Mahmood et al. 2017), random walk (Pons and Latapy 2005; Rosvall and Bergstrom 2008), and so forth. In recent years, network embedding based clustering methods have emerged, where nodes in the network are firstly represented as low-dimensional vectors and then clustered into communities, such as DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) and GraRep (Cao, Lu, and Xu 2015). Embedding based clustering methods are also used in attributed graphs (Li et al. 2018).

Label propagation methods firstly initialize the label of each node, then update the labels iteratively, and finally determine the communities by the label distribution. Typical label propagation methods include LPA (Raghavan, Albert, and Kumara 2007), SLPA (Xie, Szymanski, and Liu 2011), NMLPA (Huang, Wang, and Wang 2019), and so on.

Community detection that takes additional factors into account other than the topological structure is also a research hotspot, such as node attributes (Yang, McAuley, and Leskovec 2013), incomplete networks (Xin et al. 2017) and link semantics (Jin et al. 2018). However, it is hard to handle all the community structures in a large scale network, especially in some online and dynamic systems. Community search, i.e., trying to discover local community structures, is more practical and efficient.

k-core based FORTE

A k-core is the biggest subgraph of a network such that each of its nodes has a degree no less than k (Cui et al. 2014). On the basis of k-core, the definition of forbidden nodes aware community search based on k-core is given in Definition 1, where $AvgDist(v, S)$ denotes the average of the shortest path length from v to each node in S, and $G[S]$ denotes the induced subgraph of S in G.

Definition 1 (Forbidden nodes aware community search based on k-core)**.** *Given a graph $G = (V, E)$, a query node set Q, a forbidden node set F and a parameter k. $Q \neq \emptyset$ and $Q \cap F = \emptyset$. Find a connected induced subgraph $H = G[S]$ that contains Q without F such that the minimum degree of H is not less than k and $\forall v \in S - Q, AvgDist(v, Q) < AvgDist(v, F)$.*

As shown in Algorithm 1, a k-core based forbidden nodes aware community search (k-core based FORTE) algorithm is proposed. At first, the network is shrunk by removing forbidden nodes and related edges. Then, the Steiner tree that contains the query nodes is found to guarantee the connectivity. Next, the current community C is extended through its neighbors. Nodes are firstly sorted in descending order by their number of links to C, and then by their degrees. The one at the top is preferentially selected into the community. Considering that nodes in the community should be more close to query nodes than forbidden nodes, the nodes with longer average shortest-path length to the query nodes than forbidden nodes are pruned out. Finally, if there is no proper neighbor node that can be added and the minimum degree of C is not large enough, a global search procedure is used to guarantee the validity (Cui et al. 2014), i.e., iteratively deleting the nodes with degrees less than k until a k-core is found or query nodes have to be deleted.

Algorithm 1 k-core based FORTE

```
Require: $G = (V, E)$, $Q$, $F$, $k$
Ensure: A $k$-core community containing $Q$ without $F$.
$G \leftarrow G - H[F]$;
Calculate the Steiner Tree $T$ built on $Q$;
if $T$ cannot be found then
    return $\emptyset$;
end if
$C \leftarrow$ nodes in $T$;
mark nodes in $C$ as visited;
add unvisited adjacent nodes of $C$ into Candidates;
while $minDegree(C) < k$ do
    if Candidates is empty then
        $C = \emptyset$;
        break;
    end if
    Candidates' $\leftarrow$ nodes with most links to $C$ in Candidates;
    $p \leftarrow$ node with the largest degree in Candidates';
    if $Degree(p) < k$ then
        remove $p$ from Candidates;
    else if $AvgDist(p, Q) < AvgDist(p, F)$ then
        add $p$ into $C$;
        add unvisited adjacent nodes of $p$ to Candidates;
    end if
    mark $p$ as visited;
end while
if $C == \emptyset$ then
    return global search result of $Q$ on $G$;
else
    return $G[C]$ //$G[C]$ is the induced subgraph of $C$;
end if
```

The time complexity of Algorithm 1 depends on three parts. The first part is the shrinkage of the input network, whose time complexity is $O(n_f)$ where n_f is the number of nodes in the forbidden set. The second part is finding the Steiner tree which can be done approximately in $O(n_q n_r^2)$ where n_q and n_r are the numbers of nodes in Q and the nodes in the rest graph (Robins and Zelikovsky 2000). The last part is the extension of the current community C, whose time complexity is $O(m' + n')$ where m' is the number of edges and n' is the number of nodes in C.

k-truss based FORTE

A k-truss is a subgraph of the given network such that each of its edges has joined in no less than $k-2$ triangles (Huang et al. 2014). Based on the concept of k-truss, Huang et

Algorithm 2 k-truss based FORTE

```
Require: $G = (V, E), Q, F$
Ensure: A $k$-truss community contains $Q$ without $F$ with largest $k$.
 1: $G \leftarrow G - H[F]$;
 2: modify the trussness of each related edge after the shrinkage;
 3: $C \leftarrow$ FindG$_0$; // see (Huang et al. 2015)
 4: for each $v \in C$ do
 5:     if AvgDist$(v, Q) \geq$ AvgDist$(v, F)$ then
 6:         remove $v$ from $C$;
 7:     end if
 8: end for
 9: modify the structure of $C$ if necessary;
10: return $C$;
```

al. provide an algorithm called *FindG*$_0$ to obtain a connected k-truss containing the query node set Q with the largest k (Huang et al. 2015).

On the basis of k-truss, the definition of forbidden nodes aware community search based on k-truss is defined as follows.

Definition 2 (Forbidden nodes aware community search based on k-truss)**.** *Given a graph $G = (V, E)$, a query node set Q and a forbidden node set F. $Q \neq \emptyset$ and $Q \cap F = \emptyset$. Find a connected subgraph $H = G[S]$ containing Q without F such that H is a k-truss with the largest k and $\forall v \in S - Q, \mathit{AvgDist}(v, Q) < \mathit{AvgDist}(v, F)$.*

A k-truss based forbidden nodes aware community search (k-truss based FORTE) algorithm is proposed in Algorithm 2. The network is shrunk by removing forbidden nodes at first. Different from Algorithm 1, the trussness of each related edge has to be modified since they are used in the following steps and affect the correctness of the results. Then a k-truss community search step, such as the popular method *FindG*$_0$, is carried out. Next, the nodes in C are checked by the comparison of the average shortest-path length to the forbidden nodes and that to the query nodes, which aims to make the members in C more close to the query nodes than to the forbidden nodes. Finally, modify the structure of C by removing the edges with smallest number of triangles to maintain the k-truss if necessary.

The time complexity of Algorithm 2 can be divided into four parts. The first part takes $O(n_f)$ time complexity where n_f is the number of nodes in the forbidden set. The second is to modify the trussness of edges, which might cost $O(m)$ in the worst case where m is the number of edges in G. The third part, i.e., running *FindG*$_0$, is $O(m')$ where m' is the number of edges in C. The last part that checks the nodes in C costs $O(n')$, where n' is the number of nodes in C and there is a possible cost $O(m')$ to modify the structure of C.

The correctness of k-core based FORTE and k-truss based FORTE is obvious. For the k-core based FORTE, let $H = G[S]$ be the final resultant community of Algorithm 1. $S \cap F = \emptyset$ since S is found on $G[V - F]$. Besides, Line 9 of Algorithm 1 guarantees the minimum degree, and Line 18 of Algorithm 1 guarantees the constraint on the shortest path length. Thus, H satisfies Definition 1. For the k-truss based FORTE, the k-truss structure with the largest k is guaranteed by Line 3 of Algorithm 2 while the other two constraints on forbidden nodes can be found in Line 1 and Line 5.

The two methods based on k-core and k-truss take the average shortest path length to the query nodes and that to forbidden nodes as the measure of how close a node is to the two types of nodes. However, this measurement cannot fully exploit the influences of Q and F. For example, a node v with $\mathit{AvgDist}(v, F) = 2$ may connect to three forbidden nodes in 1-hop and one forbidden node in 5-hops. Another node p may just connect to four forbidden nodes in 2-hops, and then $\mathit{AvgDist}(p, F) = 2$. It means that the relation between p and F is hard to be differentiated from the relation between v and F though they are influenced by F differently.

Weighted Conductance based FORTE

To fully consider the influences of both Q and F, a convincing measurement should be carefully designed. That is how to measure the impact of influence. This kind of measurement can be a definite value for each node so as to decide whether it should join the community. Personalized PageRank (PPR) is a popular way to measure the proximity of nodes in a network. It is widely used in Web page rankings, which provides a numerical value to describe how important the current page is to the given pages. Thus, we put forward a novel node weighting approach based on Personalized PageRank to describe the influences of Q and F.

Weighting based on Personalized PageRank

Personalized PageRank inherits the idea of the classic PageRank algorithm, simulating a user's behavior of randomly accessing the nodes when some nodes are preferred by the user. It uses the links to recursively calculate the weight of each node, which can be interpreted as the random accessing probability. The underlying mathematical model is to assume that when a user starts from a certain node, he might jump to a set of preferred nodes with probability α, or move to one of the neighbor nodes through the adjacent link. Formally, the vector of access probabilities, denoted as $\boldsymbol{p}$, is the solution to the following equation

$$\boldsymbol{p} = \alpha \boldsymbol{v} + (1 - \alpha) \boldsymbol{M} \boldsymbol{p} \tag{1}$$

where α is the teleport probability, and is generally set to 0.15. $\boldsymbol{v}$ is called the Personalized Pagerank Vector (PPV), which is a unit vector. PPV has the same dimensions as the number of nodes, and only the values corresponding to those preferred nodes are set to non-zero. $\boldsymbol{M}$ denotes the transition matrix, which is often calculated by adjacent matrix A and diagonal degree matrix D using $M = A^T D^{-1}$.

To be fair, when setting the node weights, the forbidden nodes and the query nodes are used as preferred nodes in PPR respectively. The weight of each node is determined by normalizing the difference of two probabilities obtained through PPR. The reason to normalize the weights is that the PPR value of each node is quite small when the network is in large scale, and the differences among nodes can be seen clearly after the normalization. Next, a formal description of the node weighting is given as follows.

$$P(v) = \frac{P_q(v)}{max_{u \in V}\{P_q(u)\}} - \frac{P_f(v)}{max_{u \in V}\{P_f(u)\}} \tag{2}$$

$$W(v) = \frac{P(v) - min_{u \in V}\{P(u)\}}{max_{u \in V}\{P(u)\} - min_{u \in V}\{P(u)\}} \tag{3}$$

$P_q(v)$ denotes the PPR value of node v when using query nodes as preferred nodes while $P_f(v)$ is obtained when using forbidden nodes. V is the whole node set of the network. $W(v)$ is the final weight of node v. In Equation 2, the PPR values of node v are divided by the maximum PPR value respectively since the two values might be in different ranges. In Equation 3, the difference of two PPR values is mapped to $[0, 1]$ by min-max normalization. The larger the $W(v)$ is, the more important the node v is to the query nodes, which means the influence of forbidden nodes is smaller.

Based on node weighting, the weight given to edge (u, v) is defined as:

$$W(u, v) = \frac{W(u)}{degree(u)} + \frac{W(v)}{degree(v)} \tag{4}$$

where *degree*(u) denotes the degree of node u. Similar to node weighting, edge weighting is able to help figure out how important an edge is to the query nodes.

C_W based FORTE

Now that we have a way of node weighting to show how they are affected by both query nodes and forbidden nodes, it is intuitive to set a threshold λ to eliminate nodes that are closely related to forbidden nodes and the remaining nodes are treated as a community. However, it ignores the fact that a good community needs to be connected and cohesive.

Then, we come up with a novel metric of community, combining the structure and the influences of query and forbidden nodes. The metric is called weighted conductance, and is formalized as the following definition.

Definition 3 (Weighted Conductance)**.** *Given a graph $G = (V, E)$ and a subgraph $H \subseteq G$. The nodes in H are denoted as S. The weighted conductance of H is:*

$$C_W(H) = \frac{\sum_{u \in S, v \in V \setminus S} W(u, v)}{2 \sum_{u, v \in S} W(u, v) + \sum_{u \in S, v \in V \setminus S} W(u, v)} \tag{5}$$

where $W(u, v)$ denotes the weight of an edge (u, v).

Different from the classic conductance which measures the fraction of edges that link the community and outside in the total edges related to the community (Yang and Leskovec 2015), weighted conductance takes the weights of edges into account. When weighted conductance $C_W(H)$ becomes smaller, the community H becomes more cohesive. This requires the community to have not only fewer edges linking inside and outside, but also smaller weights for those edges. In other words, the more important the edge is to the query nodes, the more likely it is to appear in the community.

On the basis of weighted conductance, the definition of forbidden nodes aware community search based on weighted conductance is defined as follows.

Algorithm 3 C_W based FORTE

```
Require: $G = (V, E)$, $Q$, $F$, $\lambda$
Ensure: A community contains $Q$ without $F$.
for each $v \in V$ do
    if $W(v) < \lambda$ then
        remove $v$ from $G$;
    end if
end for
Nodelist ← sort the rest nodes in descending order;
startpos ← the smallest index when Nodelist[0] ∼ Nodelist[startpos] includes $Q$;
for $i$ from startpos to Nodelist.length do
    $V_i$ ← Nodelist[0] ∼ Nodelist[$i$];
    record $C_W(G[V_i])$;
end for
return $\arg\min_{G[V_i]}\{C_W(G[V_i])\}$;
```

Definition 4 (Forbidden nodes aware community search based on weighted conductance)**.** *Given a graph $G = (V, E)$, a query node set $Q \subseteq V$, a forbidden node set $F \subseteq V$ and a threshold λ. $Q \neq \emptyset$ and $Q \cap F = \emptyset$. G has been weighted by PPR as Equations 1∼ 4. Find a connected subgraph $H \subseteq G$ that contains Q without F satisfying the following conditions: (1) The weight of each node in H is not less than λ; (2) $C_W(H)$ is minimized.*

A heuristic algorithm, called weighted conductance based forbidden nodes aware community search (C_W based FORTE), is presented in Algorithm 3. It can be divided into three steps. Firstly, prune all nodes whose weights are less than the threshold λ. Secondly, sort the rest nodes by weight in descending order. Finally, check the node list from the position of last query node to the end to find the node set $V_i = ($*Nodelist*$[0], \ldots,$ *Nodelist*$[i])$ with the minimum weighted conductance. According to Definition 3, in order to decrease the weighted conductance, it is better to put edges with large weights into the community. Due to Equation 4, edges adjacent to large weight nodes tend to have large weights. Therefore, the nodes with large weights are added preferentially.

The time complexity of C_W based FORTE is divided into three parts corresponding to the three steps. The first step takes $O(n)$ to remove the nodes with weights less than λ where n is the number of all nodes. The second step takes $O(n'logn')$ to sort the rest nodes where n' is the number of rest nodes. The final step takes $O(n'm')$ to compute the weighted conductance where m' is the number of rest edges.

As we can see in Figure 2, suppose nodes A and B are query nodes, and node C is the forbidden node. After the nodes are weighted through PPR, we use the depth of color to visualize the node weight. Nodes with larger weights have a darker color. The dotted box is the community calculated by C_W based FORTE with threshold $\lambda = 0.6$. Nodes in the dotted box tend to be darker than the outside, and the three first-order neighbors of node C are not included. It makes the resultant community a better choice for users since node C and the community are well separated.

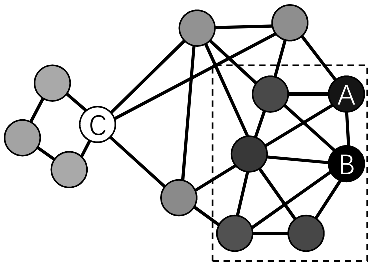


Figure 2: An example for forbidden nodes aware community search.

Table 1: Data sets

Data set	$\|V\|$	$\|E\|$	$\|C\|$
DBLP	$317,080$	$1,049,866$	$13,477$
Amazon	$334,863$	$925,872$	$75,149$
YouTube	$1,134,890$	$2,987,624$	$8,385$

Experiments

In this section, extensive experiments are conducted on real data sets to evaluate the three proposed algorithms for the problem of forbidden nodes aware community search, i.e., k-core based FORTE, k-truss based FORTE and C_W based FORTE. For convenience, they are respectively abbreviated as FORTE-k-core, FORTE-k-truss and FORTE-C_W in the following figures and tables.

Experimental setup

Table 1 lists out the data sets used in the experiments, including DBLP, Amazon and YouTube. $|V|$, $|E|$ and $|C|$ denote the number of nodes, edges and communities, respectively. The data sets are downloaded from the Stanford Large Network Data set Collection (http://snap.stanford.edu/data/). All of them have ground truth communities, which help us design reasonable test cases.

The experiments are conducted on a Server with Intel Xeon E5-2650 2.0 GHZ and 256 GB main memory. The Operation System is Windows Server 2008. All the codes are implemented using Python 3.6.1.

To evaluate the community results, we choose two suitable indicators. One is **f-measure**, which measures the accuracy of the results, and the other is **local modularity**, which measures the cohesiveness of communities.

The **f-measure** is the harmonic mean value of precision and recall. The closer the value is to 1, the closer the results are to the ground truth. Actually, the ground truth of the forbidden nodes aware community search cannot be fully known. We pick out some test cases where query nodes come from the same community C according to the ground truth offered by the data sets. When the forbidden nodes in the test cases are not in C, we treat C as the ground truth.

The **local modularity**, denoted as Q_l in Equation 6, refers to the ratio of the number of edges inside a subgraph to the total number of edges that link to the nodes in the subgraph. It describes how dense a community is in a local view. The larger value indicates a better community.

Table 2: Combinations of query and forbidden nodes

$Label$	i	ii	iii	iv	v
$\|Query\ nodes\|$	1	2	2	2	3
$\|Forbidden\ nodes\|$	2	1	2	3	2

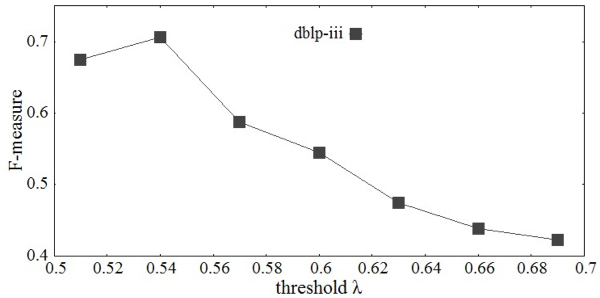


Figure 3: F-measure changes with threshold λ.

$$Q_l = \frac{k_{in}}{(k_{in} + k_{out})} \tag{6}$$

To evaluate the influence of forbidden nodes, we design a novel metric called **Closeness-to-the-Forbidden**, denoted as cf. It comprehensively reflects the probability that forbidden nodes appear in the first-order and second-order neighborhood of community members, formalized as:

$$cf(C) = 100 * \sum_{u \in C} 0.75 \frac{N1(u) \cap F}{N1(u)} + 0.25 \frac{N2(u) \cap F}{N2(u)} \tag{7}$$

where $N1(u)$ and $N2(u)$ denote the first-order neighbors and second-order neighbors of u. F is the forbidden node set. $N1(u)$ is considered to be more important so it is weighted with 0.75. The value is multiplied by 100 to make the differences among different communities more obvious. A better community should be far away from the forbidden nodes so that the fc value tends to be smaller.

By changing the number of query nodes and forbidden nodes, the following five combinations (as listed in Table 2) are used in the experiments, which are labeled with Roman numerals. For each combination, we pick 100 groups of the corresponding number of query and forbidden nodes. In each group, query nodes are randomly selected from a same community in the ground truth provided by the data sets while the forbidden nodes are randomly selected from the whole graph. Each evaluation value shown afterwards is the average of the 100 tests except for a few invalid answers when the k-core or k-truss structures cannot be found.

As for parameters, we set the threshold λ in C_W based FORTE to 0.54 according to Figure 3, which presents the f-measure changes from 0.51 to 0.69 on the DBLP data set. The parameter k in the k-core based FORTE is tested from 2 to 10, and then the largest k with valid results is remained, which varies among different tests.

Table 3: Time cost of different methods

Methods	FORTE-k-core	FORTE-k-truss	FORTE-C_W
DBLP	1.46 min	3.52 min	1.51 min
Amazon	0.99 min	7.67 min	1.41 min
YouTube	2.76 min	$>$ 10 min	2.83 min

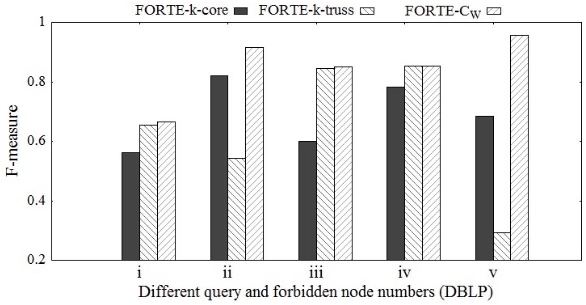


Figure 4: F-measures of different methods.

Results and analyses

In Table 3, we compare the time costs of different methods on three real networks. It is obvious that the k-truss based FORTE method has the highest time cost. The high cost mainly comes from the step of modifying the trussness of affected edges after shrinking the input network. The C_W based FORTE method is a bit slower than the k-core based FORTE method but more efficient than the k-truss based FORTE method.

As seen in Figure 4, the f-measure obtained by the C_W based FORTE method is always the best. The other two FORTE methods tend to have a lower f-measure since the community structures in the ground truth are not strictly ruled by k-core or k-truss.

As shown in Figure 5, the local modularity obtained by the k-core based FORTE method is the worst, which means the communities are not cohesive enough. The performances of the C_W based FORTE method and the k-truss based FORTE method are difficult to be compared. The C_W based FORTE method performs well in two cases while the k-truss based FORTE method performs well in three but their differences are not so distinct.

Table 4 reports the Closeness-to-the-Forbidden values of different methods. The lower the value is, the less the community is influenced by forbidden nodes. Then, we can see that k-truss based FORTE and C_W based FORTE perform better. Besides, the differences among the three methods are not so distinct, which means all of them have the ability to make nodes inside the community more close to query nodes than to forbidden nodes.

To summarize, the C_W based FORTE method performs best in f-measure. Moreover, it keeps the cohesiveness of community while reducing the influences of forbidden nodes with an acceptable time cost. That means it is a better choice to address the problem of forbidden nodes aware community search.

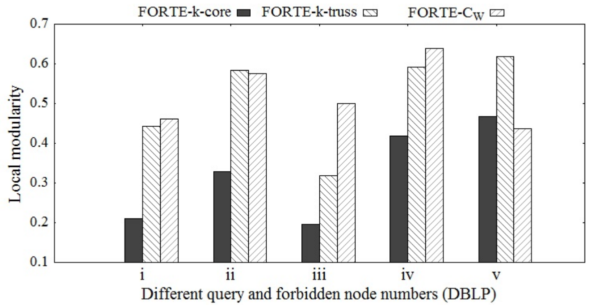


Figure 5: Local modularities of different methods.

Table 4: Closeness-to-the-Forbidden values of different methods (DBLP)

Methods	FORTE-k-core	FORTE-k-truss	FORTE-C_W
i	1.01	**0.16**	0.83
ii	1.23	**0.12**	0.16
iii	0.73	**0.01**	0.92
iv	7.21	22.8	**2.72**
v	7.20	0.62	**0.01**

Conclusion

Considering the urgent needs of users, the forbidden nodes aware community search problem is proposed in this paper. Firstly, two methods, called k-core based FORTE and k-truss based FORTE, are presented, which are based on the concepts of k-core and k-truss, respectively. Then, a novel method called C_W based FORTE is proposed, which is based on node weighting through PPR and searches for communities by minimizing weighted conductance. The experimental results demonstrate the effectiveness of the three methods.

The forbidden nodes aware community search problem indicates a new direction of community search, which adds constraints to community search from the perspective of user needs. For instance, other constraints like the community size limitation and query nodes relaxation can be taken into consideration. Further, multiple constraints are also worth exploring, which may lead to more personalized community search problems. Besides, the C_W based FORTE algorithm is shown to be effective and achieves higher accuracy, which reveals the value of solving this kind of problem. All in all, the proposed forbidden nodes aware community search problem and the related algorithms are of practical significance and may draw attention to new community search problems.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (No. 61872207), the Intelligent Manufacturing Comprehensive Standardization and New Pattern Application Project of Ministry of Industry and Information Technology (Experimental validation of key technical standards for trusted services in industrial Internet), and the China National Arts Fund (No. 20164129).

References

Akbas, E., and Zhao, P. 2017. Truss-based community search: a truss-equivalence based indexing approach. *PVLDB* 10(11):1298–1309.

Cao, S.; Lu, W.; and Xu, Q. 2015. Grarep: Learning graph representations with global structural information. In *CIKM*, 891–900. ACM.

Chen, L.; Liu, C.; Zhou, R.; Li, J.; Yang, X.; and Wang, B. 2018. Maximum co-located community search in large scale social networks. *PVLDB* 11(9).

Clauset, A.; Newman, M. E.; and Moore, C. 2004. Finding community structure in very large networks. *Physical review E* 70(6):066111.

Cui, W.; Xiao, Y.; Wang, H.; Lu, Y.; and Wang, W. 2013. Online search of overlapping communities. In *SIGMOD*, 277–288. ACM.

Cui, W.; Xiao, Y.; Wang, H.; and Wang, W. 2014. Local search of communities in large graphs. In *SIGMOD*, 991–1002. ACM.

Fang, Y.; Cheng, R.; Luo, S.; and Hu, J. 2016. Effective community search for large attributed graphs. *PVLDB* 9(12):1233–1244.

Huang, X., and Lakshmanan, L. V. 2017. Attribute-driven community search. *PVLDB* 10(9):949–960.

Huang, X.; Cheng, H.; Qin, L.; Tian, W.; and Yu, J. X. 2014. Querying k-truss community in large and dynamic graphs. In *SIGMOD*, 1311–1322. ACM.

Huang, X.; Lakshmanan, L. V.; Yu, J. X.; and Cheng, H. 2015. Approximate closest community search in networks. *PVLDB* 9(4):276–287.

Huang, B.; Wang, C.; and Wang, B. 2019. Nmlpa: Uncovering overlapping communities in attributed networks via a multi-label propagation approach. *Sensors* 19(2).

Jin, J., et al. 2015. Fast community detection by score. *The Annals of Statistics* 43(1):57–89.

Jin, D.; Wang, X.; He, R.; He, D.; Dang, J.; and Zhang, W. 2018. Robust detection of link communities in large social networks by exploiting link semantics. In *AAAI*, 314–321.

Li, Y.; Sha, C.; Huang, X.; and Zhang, Y. 2018. Community detection in attributed graphs: An embedding approach. In *AAAI*, 338–345.

Mahmood, A.; Small, M.; Al-Maadeed, S.; and Rajpoot, N. 2017. Using geodesic space density gradients for network community detection. *TKDE* 29(4):921–935.

Newman, M. E., and Girvan, M. 2004. Finding and evaluating community structure in networks. *Physical review E* 69(2):026113.

Newman, M. E. 2004. Fast algorithm for detecting community structure in networks. *Physical review E* 69(6):066133.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*, 701–710. ACM.

Pons, P., and Latapy, M. 2005. Computing communities in large networks using random walks. In *International symposium on computer and information sciences*, 284–293. Springer.

Prat-Pérez, A.; Dominguez-Sal, D.; and Larriba-Pey, J.-L. 2014. High quality, scalable and parallel community detection for large real graphs. In *Proceedings of the 23rd international conference on World wide web*, 225–236. ACM.

Radicchi, F.; Castellano, C.; Cecconi, F.; Loreto, V.; and Parisi, D. 2004. Defining and identifying communities in networks. *PNAS* 101(9):2658–2663.

Raghavan, U. N.; Albert, R.; and Kumara, S. 2007. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E* 76(3):036106.

Robins, G., and Zelikovsky, A. 2000. Improved steiner tree approximation in graphs. In *SODA*, 770–779.

Rosvall, M., and Bergstrom, C. T. 2008. Maps of random walks on complex networks reveal community structure. *PNAS* 105(4):1118–1123.

Shang, J.; Wang, C.; Wang, C.; Guo, G.; and Qian, J. 2018. An attribute-based community search method with graph refining. *The Journal of Supercomputing* 1–28. https://doi.org/10.1007/s11227-017-1976-z.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *TPAMI* 22(8):888–905.

Sozio, M., and Gionis, A. 2010. The community-search problem and how to plan a successful cocktail party. In *SIGKDD*, 939–948. ACM.

Wang, M.; Wang, C.; Yu, J. X.; and Zhang, J. 2015. Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework. *PVLDB* 8(10):998–1009.

Wu, Y.; Jin, R.; Li, J.; and Zhang, X. 2015. Robust local community detection: on free rider effect and its elimination. *PVLDB* 8(7):798–809.

Xie, J.; Szymanski, B. K.; and Liu, X. 2011. Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In *ICDMW*, 344–349. IEEE.

Xin, X.; Wang, C.; Ying, X.; and Wang, B. 2017. Deep community detection in topologically incomplete networks. *Physica A: Statistical Mechanics and its Applications* 469:342–352.

Yang, J., and Leskovec, J. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42(1):181–213.

Yang, J.; McAuley, J.; and Leskovec, J. 2013. Community detection in networks with node attributes. In *ICDM*, 1151–1156. IEEE.

Zhao, F., and Tung, A. K. 2012. Large scale cohesive subgraphs discovery for social network visual analysis. *PVLDB* 6(2):85–96.