Online Social Networks and Media 16 (2020) 100062

Contents lists available at ScienceDirect

Online Social Networks and Media

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

Efficiently discovering users connectivity with local information in online social networks

Na Li[a,*], Sajal K. Das[b]

[a] Department of Computer Science, Prairie View A&M University, Prairie View, TX USA
[b] Department of Computer Science, Missouri University of Science and Technology, Rolla, MO USA

ARTICLE INFO

Article history:
Received 20 May 2019
Revised 5 December 2019
Accepted 4 January 2020
Available online 10 March 2020

Keywords:
Online social networks
Subgraph connectivity
Search
Local view
Minimum steiner tree

ABSTRACT

People's activities in Online Social Networks (OSNs) have generated a massive volume of data to which tremendous attention has been paid in academia and industry. With such data, researchers and third-parties can analyze human beings' behaviors in social communities and develop more user-friendly services and applications to meet people's needs. However, often times, they face a big challenge of acquiring the data, as the access to such data is restricted by their collectors (e.g., Facebook and Twitter), due to various reasons, such as their user's privacy. In this paper, we intend to shed light on leveraging limited local social network topological properties to effectively and efficiently conduct search in OSNs. The problem we focus on is to discover the connectivity of a group of target users in an OSN, particularly from the perspective of a third-party analyst who does not have full access to the network. For the analyst, even discovering a user's local connections requires issuing a query through OSN APIs (e.g., Facebook Friendlist API or Twitter Followerlist API). We develop searching techniques which demand only a few number of queries for the connectivity discovery.

After conducting an intensive set of experiments on both real-world and synthetic data sets, we found that our proposed techniques perform as well as the centralized detection algorithm, which assumes the availability of the entire data set, in terms of the size of the discovered subgraph connecting all target users as well as the number of queries made in the search. The experiment results demonstrate the effectiveness of incorporating topological properties of social networks into searching in the OSNs.

1. Introduction

In the past decade, a large number of researchers have shown their interest in social networks. Particularly, they have been dedicated to designing algorithms to solve complex problems relevant to the topological structures of graphs in massive social networks, for example, community detection [1–5], detecting subgraphs with a given pattern [6] and sampling social network graphs [7]. Most of them have researched the problems assuming the availability of the entire network graph, which, however, is not realistic often times considering the API restrictions set by OSN operators for third parties to access the date they collected. Therefore, in recent years, more attention has been paid to leveraging local information and designing distributed algorithms [8–13] to solve the algorithmic issues in massive OSNs.

Our work in this paper considers the problem of discovering target users' connectivity in an OSN, particularly from the perspective of a third-party analyst who does not have a full view of the relationship graph of the social site. As a motivating example, consider that a person plans to organize a successful party/workshop, where the success is subject to three constraints: (1) a list of people must be invited to the event (i.e., target people); (2) all target participants should be acquainted with each other directly or through people who need to be invited additionally; and (3) the number of people additionally invited should be minimized due to some reasons, such as budget or space limit.

To solve this problem, we first need to find the information that can be used to measure people's acquaintance. Thanks to the development of web techniques, OSNs have attracted millions of users and collected a large amount of data from the users, including their friendship information, such as friendship on Facebook and following relationship on Twitter. Such relationship information indicates users' acquaintance and can be used to present a relationship network in the OSN.

For an OSN operator who has a view of the entire relationship network, it is easy to find out the minimum subgraph that connects all target people. In fact, similar subgraph detection problems have been studied in the domain of graph mining and graph

* Corresponding author.
E-mail addresses: nali@pvamu.edu (N. Li), sdas@mst.edu (S.K. Das).

https://doi.org/10.1016/j.osnem.2020.100062
2468-6964/

theory, varying from detecting a subgraph simply connecting a target group of people to finding a more complex one with specific restrictions on the graph size or density [14–19]. Most of the techniques developed in the literature were designed with the assumption of the availability of the entire graph. However, such an assumption is a profound limit for a third-party analyst, as they do not have the full access to the data. Therefore, the existing techniques are not applicable to our detection problem.

Solving this problem is challenging for two reasons. First, in an OSN, the information that a third-party analyst can use is limited. The analyst can gather some data either by visiting individual usersâ profile pages or by sending queries through OSN APIs. What he can see is local to the visited or queried users. Second, even discovering such local information demands effort. One can write script to crawl the web site to collect such data; however, intensively querying the OSN may cause the server to get overwhelmed. This is why many OSNs limit the number of web queries from the same IP address or a particular group of IP addresses per day. Due to the restriction, gleaning a large number of friendlists is time-consuming. Therefore, a third-party analyst needs less-cost search techniques. The contribution of this paper can be summarized as follows:

- We propose a novel subgraph detection problem for OSNs from the perspective of a third-party analyst. We intend to discover a small subgraph which covers all target users with a few number of OSN API queries.
- We design searching techniques which consist of online and offline phases to detect the desirable subgraph. Particularly, we integrate some well-known topological properties of social networks in the online searching, including small-world, power-law distribution of node degrees and the well-connectivity of high-degree nodes.
- We conduct an intensive set of experiments on both synthetic and real-world social networks to evaluate the performance of our techniques. Our finding is that the users on the OSNs are connected very well. Additionally, we can discover the connectivity of any group of arbitrarily selected nodes in an OSN with a small number of queries. Our experiment results also demonstrate the effectiveness of applying the topological properties of social networks to searching in OSNs.

The roadmap of this paper is outlined as follows. Section 2 introduces preliminaries including the topological properties of social networks, system model and problem definition. Sections 3 and 4 address the two phases of our proposed searching techniques, online searching and offline, respectively. Section 5 discusses the experimental study. Section 7 introduces the related work, followed by a conclusion in Section 8.

2. Preliminaries

2.1. Topological properties of social networks

Through many years of research in social networks, researchers have detected some important topological properties of social networks after conducting a large number of experiments and analyzing a myriad of real-world data sets. Some of these properties are well-known, like small-world, scale-free and well-connectivity among high-degree nodes.

2.1.1. Small-world property

Small-world is one of the well-known social network topological properties, which is also translated into “six degrees of separation.” It was first observed through a series of experiments conducted by Stanley Milgram and his coworkers in the 1960's [20–22]. This property causes the small diameter of social networks and ensures the existence of a short path between any pair of nodes in the social network graphs.

2.1.2. Scale-free property

A scale-free network has a power-law degree distribution, at least asymptotically. That is, the fraction $P(x)$ of nodes in the network with x direct neighbors for large values of x is given as $P(x) \sim x^{-\alpha}$, where α is a constant typically in the range $2 < \alpha < 3$. It means only a small number of nodes have very large degrees. The power-law degree distribution has been observed from many experiments over large-scale social networks.

2.1.3. Well-connectivity among high-degree nodes

Several literature, such as [23], have addressed the assortativity of social networks, indicating that nodes with similar degrees are more likely connected with each other. Particularly, the work [24] has discovered that the high degree nodes form a well-connected core from a large set of experiments on real-world data sets. In Theory 1, we prove that given two nodes, as their degrees increase, the probability of them being connected also increases. Moreover, we conducted an intensive set of experiments on real-world data sets to analyze the connectivity among nodes of high degree, which will be detailed in Section 5. Our experiment result is consistent with the finding in the work [24].

Theorem 1. *Given an undirected graph of n nodes and two nodes, v_a and v_b, with degrees d_a and d_b respectively, suppose n-1 other nodes have the same probability of connecting with v_a (v_b), then the probability of having an edge between v_a and v_b (P_{ab}) increases with d_a and d_b.*

Proof. P_{ab} can be calculated with Equation 1. The numerator shows the number of the cases where the two nodes have a connection while the denominator shows all possible cases. If the two nodes have a connection, for node a, we choose $d_a - 1$ out of $n - 2$, excluding node a and node b (i.e., $\binom{n-2}{d_a-1}$) from n nodes, and we perform the same calculation for node b (i.e., $\binom{n-2}{d_b-1}$). Their multiplication covers all cases where nodes a and b are connected.

If the two nodes do not have a connection, for node a, we choose d_a out of $n-2$ nodes, excluding node a itself and node b (i.e., $\binom{n-2}{d_a}$) from n nodes, and we do the same for node b (i.e., $\binom{n-2}{d_b}$). The multiplication of the two values gives us the possible cases when the two nodes are not linked together.

From the Eq. (1), we can see that as d_a and d_b increase, P_{ab} also increases. □

$$
\begin{aligned}
P_{ab} &= \frac{\binom{n-2}{d_a-1}\binom{n-2}{d_b-1}}{\binom{n-2}{d_a}\binom{n-2}{d_b} + \binom{n-2}{d_a-1}\binom{n-2}{d_b-1}} \\
&= \frac{1}{1 + \frac{\binom{n-2}{d_a}\binom{n-2}{d_b}}{\binom{n-2}{d_a-1}\binom{n-2}{d_b-1}}} \\
&= \frac{1}{1 + \frac{(n-d_a-1)(n-d_b-1)}{d_a d_b}} \qquad (1)
\end{aligned}
$$

2.2. System model

Although most OSNs provide all kinds of user relationship information, for example, friendship or dating relationship, we consider only friendship in this paper. Additionally, we do not quantify the strength of the friendship between users. Therefore, we can use an undirected and unweighted graph, $G(V, E)$, to model the friendship network of an OSN, where the node set V represents users and the edge set E denotes the friendships among users. Given the graph

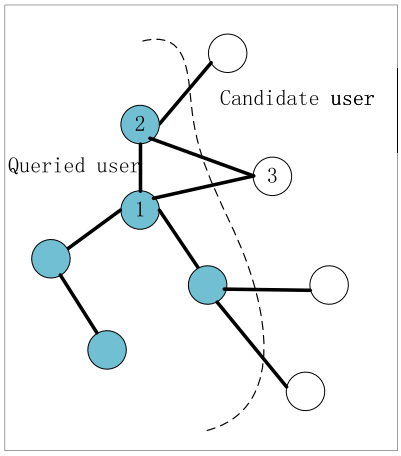

Fig. 1. The procedure of querying nodes on an OSN.

model, querying a node's friendlist can be modeled as discovering its neighboring friends which we call "local view". The number of a node's neighbors is denoted as its degree.

With the local-view discovery model, we can detect the subgraph we search for by sending a sequence of queries to the OSN. We keep track of not only nodes already queried but also a list of candidate nodes we may choose to query next. A node is called a *candidate* if it has not yet been queried but it has been discovered in the local view of a queried node. As more nodes are queried, our view in the OSN graph grows. Fig. 1 illustrates a generic querying procedure, where blue dots represent nodes we have queried while white ones are candidate nodes discovered already.

2.3. Problem definition

We name our problem Local-view based Mininum Subgraph Detection problem (LMSD), as defined in Problem 1. Note that the LMSD problem requires both the size of the detected subgraph and the number of queries be minimized, which, however, is hard to achieve at the same time. To cope with this challenge, we heuristically interpret the problem and break it down to two phases. We first conduct online search with a few number of queries to find a connected subgraph which covers all target nodes, and then in the detected subgraph we intend to discover the minimum subgraph which keeps all target nodes connected. The rationale of the effectiveness of our interpretation in solving the LMSD problem is if the number of nodes queried in the first phase is small, the size of the finally detected subgraph should not be very large.

Problem 1. Local-view based Minimum Subgraph Detection (LMSD): Given a set of target nodes S_0 in a graph, $G(V, E)$, the full topology of which is unknown initially, find the minimum number of nodes from $V \backslash S_0$ to make all target nodes connected with the minimum number of node queries for local-view discovery.

Given the subgraph discovered in the first phase, we name the minimum subgraph detection problem in the second phase the Centralized Minimum Subgraph Detection problem (CMSD). The CMSD problem is defined as given the entire graph and a group of target nodes, we look for the minimum number of extra nodes to connect all of the target nodes together. The CMSD problem is a hard problem as proved in Theorem 2 below. The complexity of the CMSD indirectly indicates the hardness of the LMSD problem, because the former is part of the latter. Based on our two-phase based interpretation, we will first discuss how to detect the connectivity of target nodes via a few number of online queries in Section 3 and then talk about algorithms to discover a smaller connected subgraph offline from the data collected in the previous phase.

Theorem 2. *The Centralized Minimum Subgraph Detection problem (CMSD) is NP-hard.*

Proof. We will prove the NP-hardness of CMSD by a reduction to the Steiner Tree problem (ST). The definition of ST is: Given an unweighted graph G and a set of nodes V_t in it, find a tree with minimum number of edges in G, which make any two nodes in V_t reachable to each other either directly or indirectly via other nodes in G. As is well known, the ST problem is NP-hard [25]. The decision version of ST is that given an unweighted graph $G(V, E)$, a set of nodes $V_t \subseteq V$ and an integer k, we are looking for a tree which involves all nodes in V_t and contains at most k edges from E. The decision version of CMSD problem is that given an unweighted graph $G'(V', E')$, a set of nodes $V_t' \subseteq V'$ and an integer k', we are searching for a subgraph of G' which includes all nodes in V_t' and covers at most k' nodes from $V' \setminus V_t'$.

Now we will demonstrate that there is a solution for ST if and only if there is a solution for CMSD. Evidently, the nodes in any steiner tree with at most k edges will be the solution of CMSD, where $k' = k + 1 - |V_t|$. On the other hand, any spanning tree of the subgraph found in CMSD will form a steiner tree with at most $k' + |V_t'| - 1$ edges. Here the spanning tree is referred to as a tree composed of all the nodes and some (or perhaps all) of the edges of a given graph. Therefore, the CMSD problem is NP-Hard. □

3. Online searching

In the online searching, we intent to discover a subgraph in the OSN to connect all target nodes with a few number of queries. The traditional graph searching techniques, such as *Depth First Search* (DFS) or *Breadth First Search* (BFS), can be applied as the brute-force subgraph detection techniques; however, their cost on individual queries is non-trivial without knowing the topology of the entire OSN graph. Therefore, we aim to design more efficient searching techniques to discover the connectivity of the targets.

3.1. The starting point of search

Without any prior knowledge, searching from the target nodes is a reasonable starting point. After we query all target nodes, each of them and its neighbors discovered through the OSN API form a subgraph. These subgraphs are most likely disjoint due to the structural sparsity of social networks. Each of these subgraphs has its own node candidate set for further queries. The candidate set of a subgraph initially contains only the neighbors of its target node, but it grows as more nodes are queried.

Given the scattered subgraphs, efficiently discovering the connectivity of target nodes requires merging all of these subgraphs quickly by querying a small number of nodes. One can see that in order to solve this problem, the selection of nodes to query is critical. In the following subsections, we will define two criteria to evaluate the importance of a node in the online searching.

3.2. The evaluation of node candidates

In a dense graph ($|E| \gg |V|$), it's straightforward to pick a good node candidate for query. Basically, a node which can make more target nodes accessible to each other should be selected. However, such a criterion is not sufficient to determine a candidate node in a sparse graph, such as social network graphs, and may even lead to the failure of the search process. The reason is that in a sparse graph more likely none of the node candidates can directly improve the reachability of target nodes. Therefore, we need a new

criterion to evaluate a node's capability of merging the subgraphs associated with target nodes.

Inspired by the critical role of high-degree nodes in searching on social network graphs [26,27], we prioritize node candidates of high degree for query in the online searching. However, often times, the real degree of a node candidate is unknown until we query it. Considering the query restriction, we propose two approaches to estimate the degrees of node candidates.

(1) Pre-Degree. A node's pre-degree is the number of the node's neighbors which have been discovered in online searching upon to a time point. An example is illustrated in Fig. 1. In this snapshot of our search, the pre-degree of node 3 is two at that time point, as we only see its connections with node 1 and node 2. As more nodes are queried, we may discover more connections of node 3, which causes its pre-degree to increase accordingly.

We use a node's pre-degree to estimate its real degree. The rationale is that since social network graphs have power-law degree distributions, if we see a node has a very high pre-degree, the real degree of that node will probably be high as well. However, this may not always be accurate when a node's pre-degree is low.

(2) Creation Time. The time a user created his/her account in the OSN is also useful to infer its real degree in the OSN graph. The rationale came from the Barabasi-Albert(BA) model [28], which is a well-known model for generating random scale-free networks using a preferential attachment mechanism. In this model, new nodes are added to the network one at a time. Each new node is connected to existing nodes with a probability that is proportional to the number of links that the existing nodes already have. This tells us it's highly likely that the earlier a node (e.g., a user account) was created in the social network, the higher degree the node has, which is also addressed in the paper [29].

In most of the OSNs, although the account creation time is available on the user's profile web page, we have to issue a query to retrieve that information. However, we notice that on some OSNs, like Twitter, users' numeric IDs are assigned sequentially. A smaller user ID indicates the earlier creation time of that user's account. Therefore, from the follower list returned from the Twitter API, we can see which candidate node was created earlier so as to prioritize it for query.

3.3. Algorithmic techniques for online search

We propose two online search techniques for detecting the connectivity of target nodes with a few number of queries, called *Unbalanced Multiple-Subgraph Searching (UMS)*, and *Balanced Multiple-Subgraph Searching (BMS)*, respectively. We break each query step down into two phases, first to decide which target node's subgraph to choose and second to decide which node from the candidate set of the subgraph to query. The difference of the two techniques is in the subgraph selection. In order to evaluate a subgraph, we define *subgraph degree* in Eq. (2) as the maximum (estimated) degree of nodes in the subgraph, where the nodes include not only already queried nodes but also the ones in the candidate set, and $D(u)$ represents the estimated degree of u in the subgraph.

$$DSub(i) = max(\{D(u)|u \in subgraph(i)\}) \quad (2)$$

3.3.1. Unbalanced Multiple-Subgraph Search (UMS)

The basic idea of UMS is to prioritize not only high-degree nodes but also high-degree subgraph to query in the online search. The UMS technique consists of three steps: (1) query all of the target nodes in the OSN graph and form their subgraphs individually; (2) select the subgraph with maximum subgraph degree as the *target subgraph*; and (3) query the node in the candidate set of the target subgraph which has the largest degree. The node degree can be estimated in terms of either of the two criteria, the pre-degree or the creation time, as we discussed earlier. A tie will be broken arbitrarily. The Steps 2 and 3 will be repeated until the subgraphs of all target nodes are merged together.

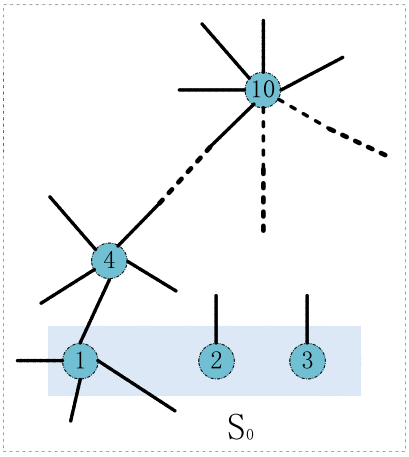

Fig. 2. An example of the inefficiency of UMS.

Note that after querying a node, the target subgraph and its node candidate set will be updated according to the list of newly discovered nodes returned from the query. If the query causes any overlap between the target subgraph and another subgraph, they will be merged together. Specifically, their sets of node candidates and of nodes already queried will be merged respectively. We call this scheme *Unbalanced* Multiple-Subgraph Search as we realize that once the target subgraph is determined at the beginning of the search, it will never be changed. This observation is proved in Property 1.

Property 1. *With UMS, the target subgraph will never be changed once it is picked at the beginning of the search.*

Proof. Based on the definition of the target subgraph, a subgraph is selected as the target subgraph if it has the maximum degree, which is evaluated by the largest degree of the nodes in that subgraph. After we query one node, the largest degree can only be increased, regardless of whether it's evaluated by the pre-degree or the creation time (or user id). Therefore, the previously picked target subgraph will still have the largest degree among all of the subgraphs, thereby always being the target subgraph. □

An example is illustrated in Fig. 3. After querying nodes, v_1, v_2 and v_3, three subgraphs, $Subg_1$, $Subg_2$ and $Subg_3$, are correspondingly formed with degrees two, one and three, separately. Based on the maximum degree rule in the UMS searching, $Subg_3$ is chosen as the target subgraph. v_4 is randomly selected as the first node to query from the candidate set of $Subg_3$ as there are three node candidates having the same degree. Then, v_5 becomes the candidate with the highest pre-degree in $Subg_3$, therefore, we query v_5, which leads to the merging of $Subg_2$ with $Subg_3$. The search continues by querying nodes selected from the candidate set of $Subg_3$ until $Subg_1$ is also merged with $Subg_3$.

3.3.2. Balanced Multiple-Subgraph (BMS)

The inspiration in designing BMS came from our concern over the efficiency of searching with UMS. One can see that essentially UMS prioritizes high-degree nodes in the search, which may not be able to efficiently reach out to the target nodes of low degrees. For example, in Fig. 2 the subgraphs initialized with the low-degree target nodes can do nothing but waiting the target subgraph to reach out to them. However, since the high-degree nodes in social networks are well connected as we introduced in Section 2, if we

could reduce the degree difference among the subgraphs by prioritizing the subgraphs of low degrees in searching, the procedure of merging subgraphs may perform faster.

Algorithm 1: The Framework of Our Online Searching Techniques

```
Input: Set of target nodes, TS, and an oracle of querying
       nodes in the OSN
Output: A connected subgraph covering all nodes in TS
foreach Node v_i in TS do
    subgraph(i) = Query(v_i);
    list.add(subgraph(i));
while list.size != 1 do
    tsg = SelectTargetSubgraph(list);
    tn = SelectNode(tsg);
    subgraph(tsg).add(Query(tn));
    if CheckOverlap(list, tsg) then
        Merge(list, tsg);
    foreach Node candidate v_m in tsg do
        Update(D(v_m));
```

Algorithm 2: Selection Target Subgraph for UMS

```
Input: List of subgraphs sg_1, sg_2, ..., list
Output: The target subgraph
tsg = sg_1;
foreach sg_i in list do
    if DSub(sg_i) > DSub(tsg) then
        tsg = sg_i;
return tsg;
```

Algorithm 3: Selection Node For UMS and BMS

```
Input: Target Subgraph, tsg
Output: The node to query
NC = tsg.nodecandidates;
tn = v_0 in NC;
foreach Node i in NC do
    if D(i) > D(tn) then
        tn = v_i;
return tn;
```

Inspired by this idea, we design the BMS search scheme which also consists of three steps: (1) query all target nodes in the OSN graph and form individual subgraphs; (2) select the subgraph with the minimum subgraph degree as the *target subgraph*; and (3) query the highest-degree node from the candidate set of the target subgraph (break ties arbitrarily). The Steps 2 and 3 will be repeated until the subgraphs of all target nodes are merged together. Similar to UMS, if a query with BMS causes any subgraphs to overlap, they will be merged and the degree of the target subgraph will be updated accordingly. Therefore, the previously picked target subgraph may not be selected for the next query if its degree does not remain the minimum among all the subgraphs.

Unlike UMS, BMS focuses on low-degree subgraphs in our searching. Therefore, we call this technique Balanced Multiple-Subgraph Searching. Let us run BMS on the simple example we used before for UMS, as shown in Fig. 4. Initially, the subgraphs

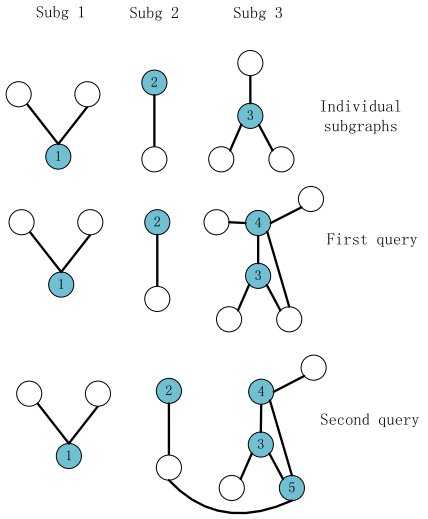

Fig. 3. The example of using UMS technique.

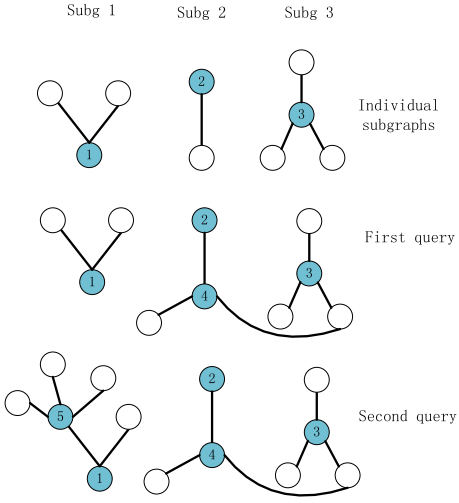

Fig. 4. The example of using BMS technique.

formed by querying target nodes have degrees two, one and three. Since $Subg_2$ has the minimum degree, it is defined as the target subgraph in the first query. Then the only node candidate in $Subg_2$, v_4, is queried, which causes the merging of $Subg_2$ and $Subg_3$. The newly merged subgraph has degree of three, which is larger than the degree of $Subg_1$, therefore, the target subgraph is reassigned with $Subg_1$. In $Subg_1$, v_5 is selected to query, which grows $Subg_1$ and increases its degree to four. At this point, the target subgraph will be reassigned again. The search continues until all subgraphs are merged.

Algorithm 4: Selection Target Subgraph for BMS

```
Input: List of subgraphs sg_1, sg_2, ..., list
Output: The target subgraph
tsg = sg_1;
foreach sg_i in list do
    if DSub(sg_i) < DSub(tsg) then
        tsg = sg_i;
return tsg;
```

Here we want to emphasize the unique topological properties of social networks used in design of BMS, which ensure the effi-

ciency of merging the subgraphs of target nodes in the OSN. First of all, based on the literature [27], we know that high-degree node based search can reach nodes of the highest degree with about $O(n^{\frac{\gamma-2}{\gamma-1}})$ steps in social network graphs, where γ is the exponent of the power-law distribution, and n is the number of nodes in the networks. Therefore, in our search, each subgraph will reach node of the highest degree with a few steps of searching. Second, because of the limited number of nodes of the highest degrees in scale-free social networks, searching multiple subgraphs along high-degree nodes will let the subgraphs grow towards the nodes of the highest degrees and merge together. Third, due to the well-connectivity among high-degree nodes, balancing the search to prioritize low-degree subgraphs will speed up the merging of subgraphs associated with target nodes. Incorporating all of these topological properties in design of BMS ensures its efficiency of searching.

4. Offline detection

In the offline detection phase, we aim to find a smaller subgraph from the subgraph discovered in the online searching which can maintain the connectivity of all target nodes. Considering the association between the CMSD problem and the Steiner Tree problem (ST) as we discussed in Section 2, we apply a classic approximate ST algorithm [30] to detect a smaller subgraph in the offline detection phase.

There are two main reasons for us to apply the ST algorithm [30]. First, it can guarantee the size of the detected subgraph is no larger than $2(1-1/\ell)$ times the size of the optimal subgraph, where ℓ is the number of leaves in the optimal tree. Second, it runs faster with the time complexity $|S_0||V|^2$, which is a critical concern when running algorithms on large-scale OSN data sets.

Given an undirected and unweighted graph $G(V, E)$ and a set of target nodes $S_0 \subseteq V$, there are four steps to find a heuristic minimum steiner tree in [30]: (1) construct the complete undirected graph $G_1(V_1, E_1)$ by creating an edge between each pair of nodes in S_0 with a label of the length of their shortest path on G; (2) find the minimal spanning tree T_1 of G_1; (3) construct a subgraph G_s of G by replacing each edge in T_1 by its corresponding shortest path in G; and (4) find the minimal spanning tree T_s of G_s. Delete from T_s edges with leaves which are non-steiner points.

5. Experimental study

To evaluate the performance of our techniques in solving the LMSD problem, we conducted experiments not only on large-scale real-world data sets but also on a synthetic data set. In the following subsections, we will first introduce the data sets used in the experiments and analyze their topological properties, including the degree distribution and the connectivity of high degree nodes. Then, we will evaluate the two steps in making each query with our techniques – picking a target subgraph first and then choosing a node to query.

Additionally, we implemented a variation of Breath First Search (BFS) [31,32]: place all targets in a queue first, and then query each node in the queue and enqueue its neighbors accordingly until all targets are reachable to each other. The BFS was proposed in [31,32] to crawl the OSN to collect data for OSN analysis, such as estimating any user property and some topological properties. Although BFS was not particularly designed to solve our problem, since it's a well-know approach to collect data in OSNs, we tailored it and use it as a benchmark for comparison purpose.

Table 1
Largest component from data sets for our experiments.

Date Sets	Nodes	Edges	LC(V, E)	C
Slashdot	82168	504230	(82168,504230)	1
Gowalla	196591	950327	(196591,950327)	1
Brightkite	58228	214078	(56739,212945)	547
Facebook	63731	817090	(63392,816886)	144
Synthetic	80000	1999375	(80000,1999375)	1

5.1. Data sets

We used four real-world data sets and one synthetic data set in our experimental study. All real-world data sets excluding Facebook [33] can be downloaded from the repository [34].

(1) Facebook data set [33]: The data was crawled from Facebook.com, capturing the friendship between users, which can be modeled as an undirected graph.

(2) Slashdot data set [35]: The data contains the friend/foe links between the users of Slashdot. The data set does not distinguish friendship from foeship between users. The links in the original set are directional, We converted the Slashdot data set to an undirected graph for our experimental study. Specifically, if there is originally one edge between two nodes, regardless of their direction, we correspondingly create an edge between the nodes in the undirected graph.

(3) Gowalla data set [36]: Gowalla is a location-based social networking web site where users share their locations by checking in. The data collected from Gowalla present the friendship network which is undirected.

(4) Brightkite data set [36]: Brightkite was once a location-based social networking service provider where users shared their locations by checking-in. The friendship network is originally directed, but we have constructed a network with undirected edges whenever there is a friendship regardless of the direction.

(5) Synthetic data set: We generated a random graph using the Barabasi-Albert preferential attachment model [28]. In the model, a graph of n nodes is grown by attaching new nodes each with m edges that are preferentially attached to existing nodes with high degree. We set $n = 80000$ and $m = 25$ to generate our synthetic set which has a size similar to the size of Facebook data set.

As we study on how to connect a group of nodes together on an OSN, we need to ensure all the target nodes are indeed reachable to each other in the OSN graphs, which means the undirected input graphs should be connected. Therefore, we preprocessed the original data sets by extracting the largest connected component from each of them. In Table 1, we list the numbers of edges, nodes, and components (i.e., C) as well as the size of largest component (i.e., $LC(V, E)$) in each original data set. In our experiments we used the largest connected components as the input graphs for evaluating our algorithms.

5.2. Topological properties of data sets

We examined some topological properties of our data sets which we introduced in Section 2, including power-law degree distribution and the well connectivity of high-degree nodes.

5.2.1. Power-law degree distribution

We applied the statistical framework for discerning and quantifying power-law behavior in empirical data proposed in [37] to check the degree distribution of our data sets. We ran the framework program [38] on our data sets. In power-law distribution, $P(x) \sim x^{-\alpha}$, α is known as the exponent or scaling parameter, which typically lies in the range $2 < \alpha < 3$. More often the power law applies only for values greater than some minimum *xmin*. In such cases we usually say that the tail of the distribution follows

Table 2
Fitting the power-law distribution to empirical data.

Date Sets	Slashdot	Gowalla	Brightkite	Facebook	Synthetic
alpha	3.46	2.83	2.56	4.44	2.99
xmin	219	95	24	157	47

Table 3
Subgraphs of high-degree nodes.

≥ **Degree**		100	200	300	400	500	600
Facebook	**nodes**	3307	461	106	46	26	11
	Components	1	1	1	1	1	2
	Average Distance	2.61	2.26	1.98	1.68	1.70	1.58
Slashdot	**nodes**	1916	757	235	115	61	39
	Components	1	2	2	1	3	3
	Average Distance	2.11	2.0	1.91	1.92	2.01	2.26
Gowalla	**nodes**	1787	494	245	143	99	77
	Components	1	1	1	1	1	1
	Average Distance	2.32	1.96	1.82	1.70	1.63	1.58
Brightkite	**nodes**	408	89	30	14	9	7
	Components	1	1	1	1	1	1
	Average Distance	2.29	1.95	1.88	1.78	1.67	1.71
Synthetic	**nodes**	5181	1348	592	348	240	169
	Components	1	1	1	1	1	2
	Average Distance	2.61	2.26	1.98	1.68	1.70	1.58

a power law. Therefore, we checked the values of the two parameters, α and *xmin*, for all of our data sets and the results are listed in Table 2.

5.2.2. Connectivity of high-degree nodes

To evaluate the connectivity of high-degree nodes in our data sets, we first extracted the subgraph formed by nodes with a degree more than a threshold and the edges among them. The threshold ranges from 100 to 600 in increments of 100. We analyzed the number of the connected components and the average length of the shortest paths (i.e., distance) between any pair of reachable nodes in each extracted subgraph. In calculating the average distance, if an extracted subgraph has more than one component, we calculated the average distance in each component first and then average them.

From Table 3, we can see that although the number of nodes decreases as the degree threshold goes up, the nodes of high degree are still connected well. Furthermore, the average distance between any reachable pair of nodes is about 2, as shown in Table 3. These results demonstrate the well-connectivity among high-degree nodes in our OSN data sets.

5.3. The evaluation of techniques

We conducted multiple groups of experiments with each data set by varying the number of selected target nodes, ranging from 20 to 100 in increments of 20. Furthermore, given a specific number of target nodes, we ran 100 rounds of experiments by selecting target nodes uniformly at random. The same set of targets were used to compared different strategies. As we discussed in Section 3, there are two steps in the online searching, choosing the target subgraph first and then selecting a node to query. Therefore, we evaluate these two steps, respectively. In addition, we compared one of our searching techniques with two other landmark solutions for graph searching. One is a variation of Breadth First Search (BFS) which starts from multiple target nodes, and the other is to randomly choose the target subgraph first and then randomly select a node from the subgraph to query, which is therefore called DoubleRandom solution.

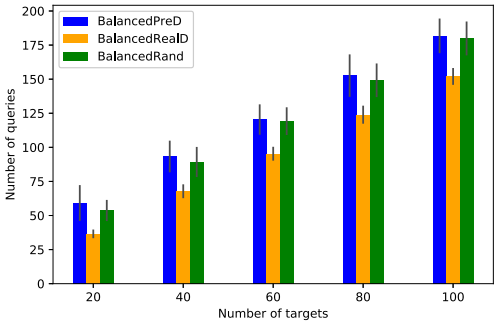

Fig. 5. Facebook: Queries - NS.

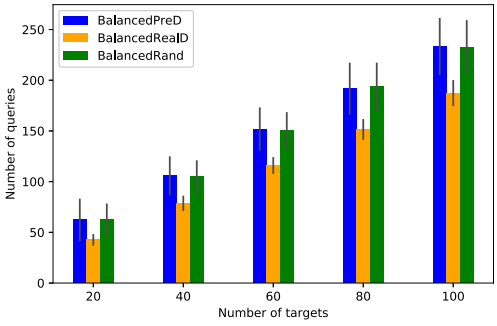

Fig. 6. Slashdot: Queries - NS.

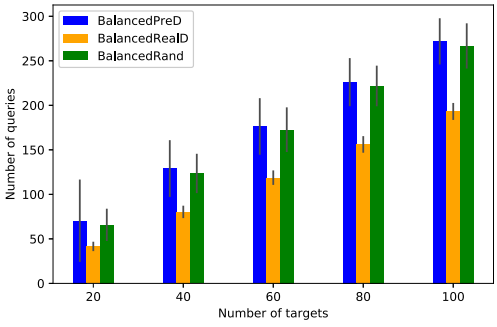

Fig. 7. Gowalla: Queries - NS.

5.3.1. Node selection (NS)

In this group of experiments, we used the balanced subgraph selection strategy which targets the subgraph with the lowest degree and evaluated different node selection strategies, including pre-degree based strategy (BalancedPreD), real-degree based strategy (BalancedRealD) as well as the strategy of randomly selecting next node for query (BalancedRand). Additionally, for the synthetic data set, we implemented the creation-time based strategy (BalancedCreaT). Since the synthetic data set was generated with the Barabasi-Albert preferential attachment model [28], the nodes with smaller ids were created earlier. Therefore, the node ids signify the order of user account creations, which is the other way we proposed in Section 3 to estimate node degrees. We evaluated these node selection strategies in terms of the number of online queries and the number of extra nodes selected in the offline detection for reaching nodes connectivity. We validated whether high-degree nodes are good choice for search and verified the goodness of using pre-degree and creation time to estimate the real degree of a node candidate.

In terms of the number of queries displayed in Figs. 5–9, we can see: (1) BalancedRealD outperforms the others. In the real world, some online social networks, like Linkedin.com, do provide the number of connections of neighboring nodes without additional

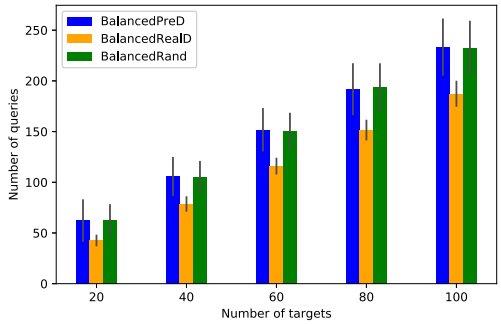

Fig. 8. Brightkite: Queries - NS.

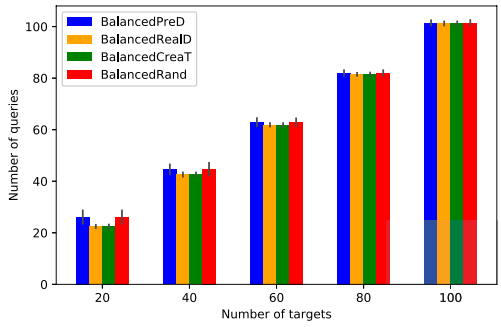

Fig. 9. Synthetic: Queries - NS.

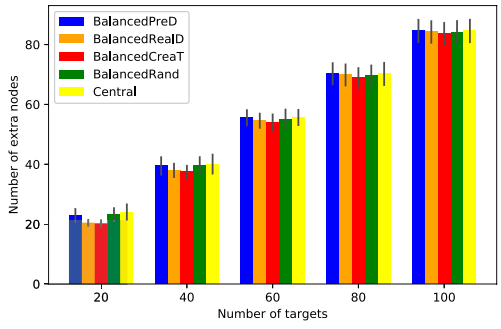

Fig. 10. Synthetic: Extra Nodes - NS.

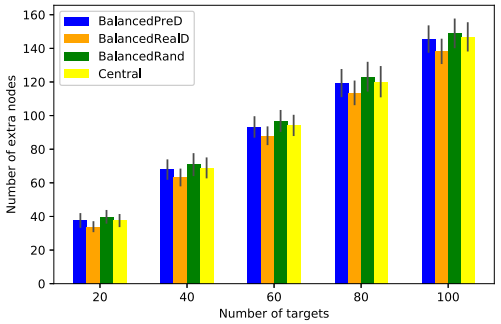

Fig. 11. Facebook: Extra Nodes - NS.

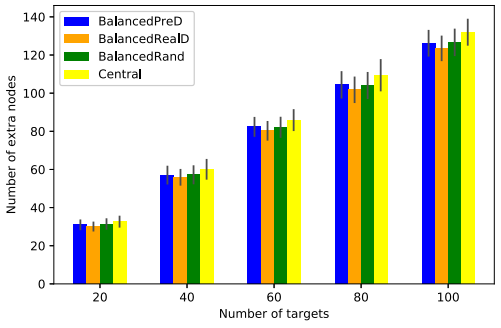

Fig. 12. Slashdot: Extra Nodes - NS.

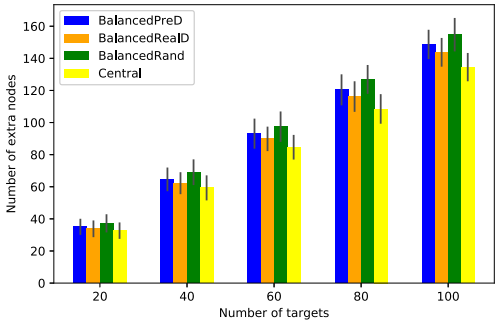

Fig. 13. Gowalla: Extra Nodes - NS.

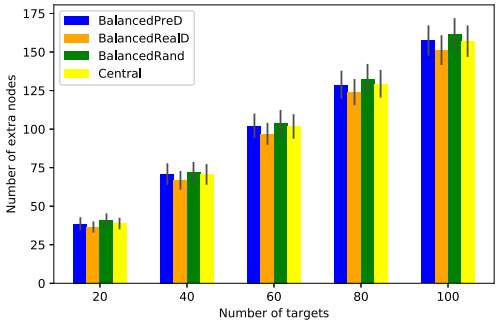

Fig. 14. Brightkite: Extra Nodes - NS.

query effort. (2) BalancedCreaT performs better than BalancedPreD, and does as well as BalancedRealD for most of the experiments. (3) BalancedPreD and BalancedRandom issued a similar number of queries in the online search. At first glance, the 3rd observation is unexpected. But we believe it is because we applied balanced subgraph search for both strategies, which predominates the search process regardless of the node selection. In order to verify this thought, we ran another group of experiments on real-world data sets with UnbalancedRand - sticking with the subgraph of highest degree but randomly selecting node to query from the subgraph. The result is that all online search failed due to not being able to achieve the targets connectivity with queries less than 10 times of the number of targets, which we set as a termination condition.

In terms of the number of extra nodes needed for connecting targets presented in Figs. 10–14, we can observe that: (1) BalancedRealD outperforms the others. (2) When comparing the local-view based strategies with Central which assumes the availability of the entire data set, their results are comparable, and sometimes, local-view based strategies perform better than Central. This is because Central is an approximate algorithm rather than the optimum one, as discussed in Section 4. (3) BalancedCreaT performs similarly to BalancedRealD in the synthetic data set. (4) BalancedPreD performs a little better than BalancedRand, needing less number of nodes for making targets reachable.

5.3.2. Subgraph selection (SS)

In order to evaluate subgraph selection techniques, we implemented the unbalanced subgraph selection (UMS), and the balanced subgraph selection (BMS). The UMS always targets the subgraph with the largest degree, while the BMS always chooses the subgraph with the lowest degree. In this group of experiments,

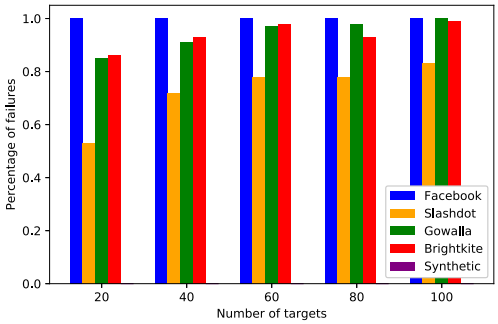

Fig. 15. Percentage of Failures - BFS.

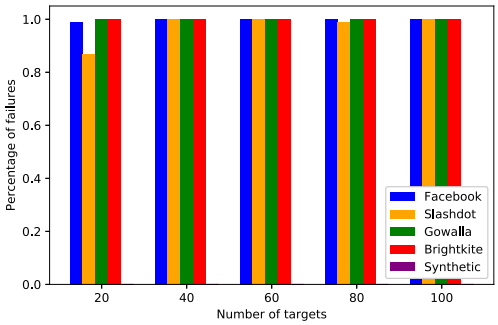

Fig. 16. Percentage of Failures - UnbalancedReal.

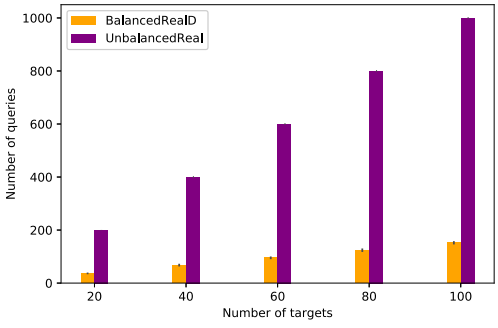

Fig. 17. Facebook: Queries - SS.

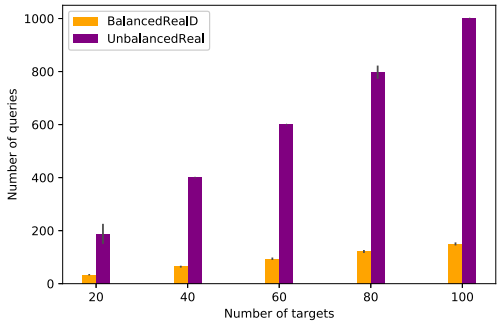

Fig. 18. Slashdot: Queries - SS.

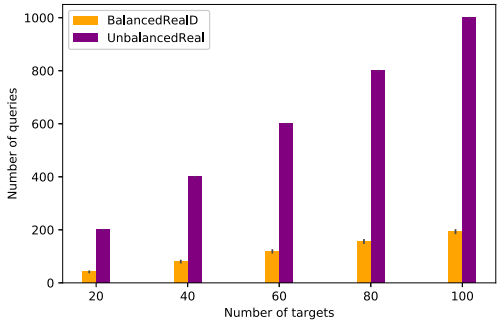

Fig. 19. Gowalla: Queries - SS.

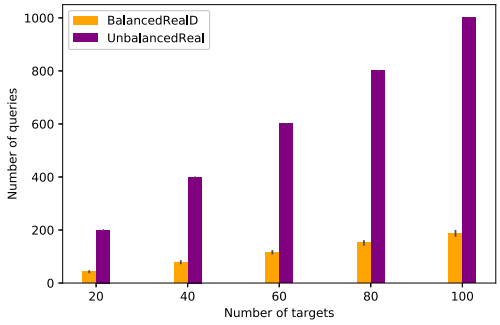

Fig. 20. Brightkite: Queries - SS.

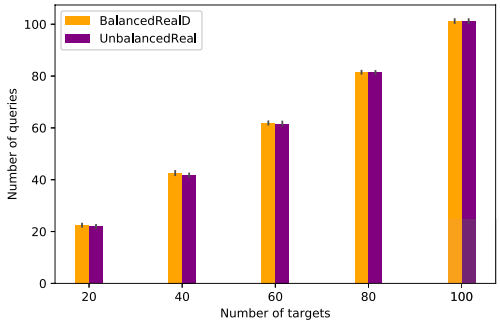

Fig. 21. Synthetic: Queries - SS.

we applied the real degrees of nodes for node selection aiming at eliminating the impact of node section on evaluating the performance of subgraph selection techniques. After running some experiments, we realized that the number of queries issued with BMS is about two times of the number of targets; however, UMS may search through a large portion of the data set to complete the online search. Therefore, for UMS, we set a termination condition: if the number of queries issued is more than 10 times of the number of targets, we will terminate the search.

In Figs. 17–21, we can see: (1) Except for the synthetic data set, BalancedRealD requires much less number of queries to reach the connectivity of targets than UnbalancedReal does. (2) For the real-world data sets, the number of queries issued with UnbalancedReal reaches almost 10 times of the number of targets. This is because most of the experiments with UnbalancedReal were terminated by the condition we set and failed to achieve the connectivity of targets as displayed in Figs. 15-16.

In Figures 22–26, we can observe: (1) For the real-world data sets, BalancedRealD performs as well as the Central, and even better in some cases. (2) Since UnbalancedReal failed to achieve connectivity in the real-world data sets, its result was not displayed except for some cases where the online searching successfully discovered the connectivity of targets. (3) For the synthetic data set, the performance of BalancedRealD and UnbalancedReal are similar.

5.3.3. Comparison with landmarks

In this group of experiments, we evaluated BalancedRealD more by comparing it with Breath First Search (BFS) and randomness based search. The BFS begins with all targets and then queries their neighbors, so on and so forth. We implemented the randomness based search as randomly selecting the target subgraph first and

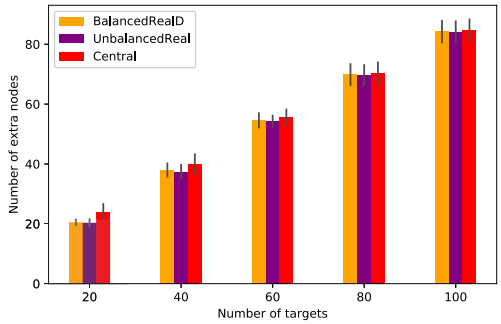

Fig. 22. Synthetic: Extra Nodes - SS.

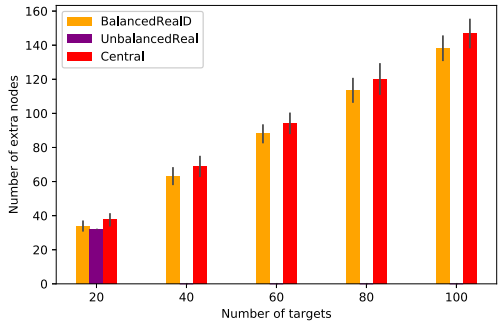

Fig. 23. Facebook: Extra Nodes - SS.

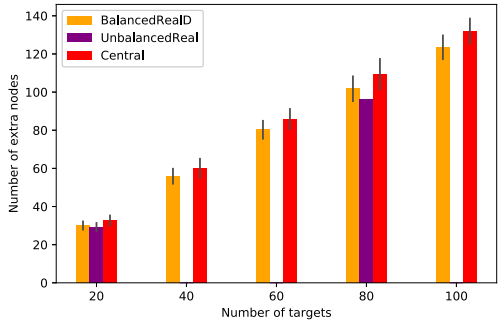

Fig. 24. Slashdot: Extra Nodes - SS.

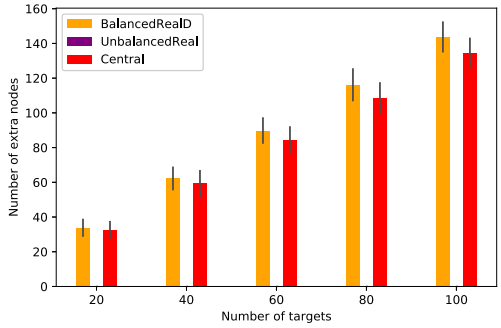

Fig. 25. Gowalla: Extra Nodes - SS.

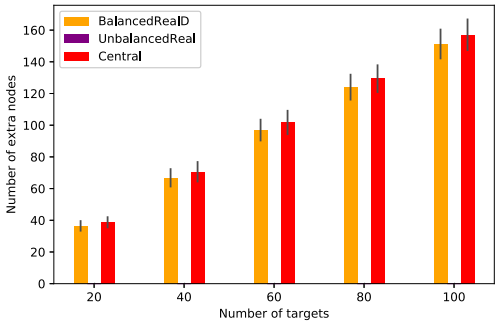

Fig. 26. Brightkite: Extra Nodes - SS.

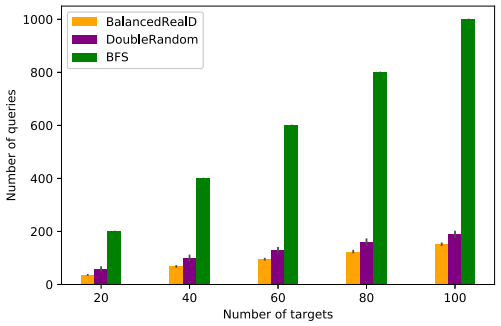

Fig. 27. Facebook: Queries - Landmark.

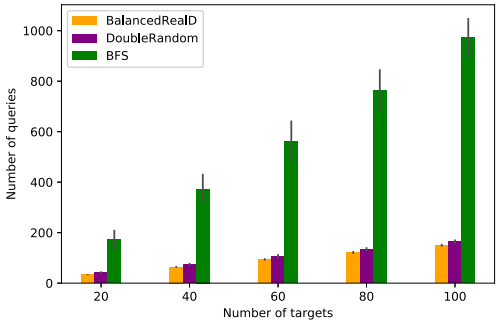

Fig. 28. Slashdot: Queries - Landmark.

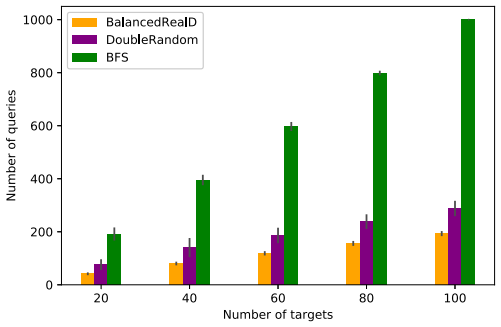

Fig. 29. Gowalla: Queries - Landmark.

then randomly choosing a node to query from the subgraph. Therefore, we name it DoubleRandom.

In terms of the queries as displayed in Figs. 27–31, BFS performs much worse than BalancedRealD and DoubleRandom, which is caused by the implementation of BFS. Specifically, if a high degree node is visited, soon the search will visit all of its neighbors. Therefore, very likely, the search will get stuck with one subgraph, causing the unbalance. For DoubleRandom, as we mentioned earlier, the subgraph selection predominates the node selection. Therefore, DoubleRandom also gives other subgraphs a chance to target, so it performs better than BFS. Another observation from the figures is that BalancedRealD issued less queries than DoubleRandom did.

For the extra nodes discovered from the offline search, as showed in Figs. 32–36, BalancedRealD requires less extra nodes than DoubleRandom does. For BFS, as displayed in Fig. 15, in most of the experiments on real-world data sets, BFS failed to achieve the connectivity of targets in the online search. Therefore, the re-

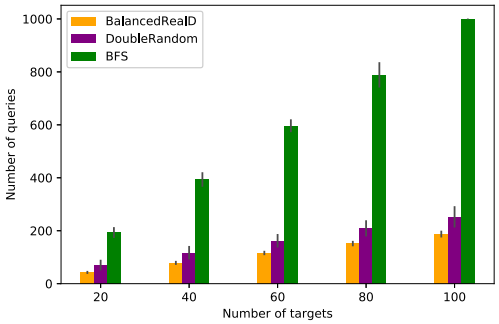

Fig. 30. Brightkite: Queries - Landmark.

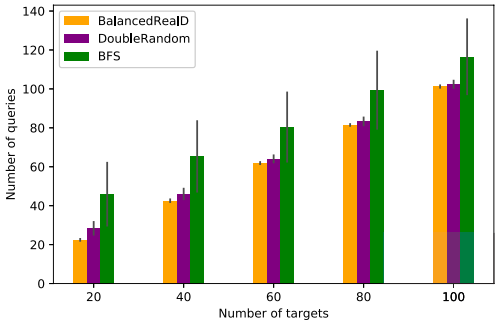

Fig. 31. Synthetic: Queries - Landmark.

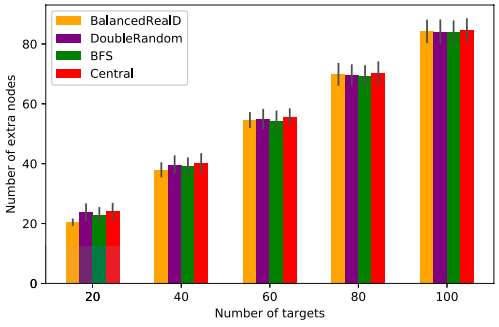

Fig. 32. Synthetic: Extra Nodes - Landmark.

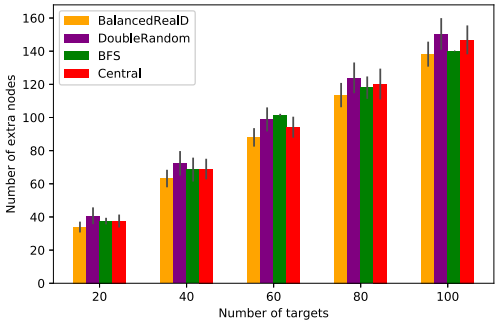

Fig. 33. Facebook: Extra Nodes - Landmark.

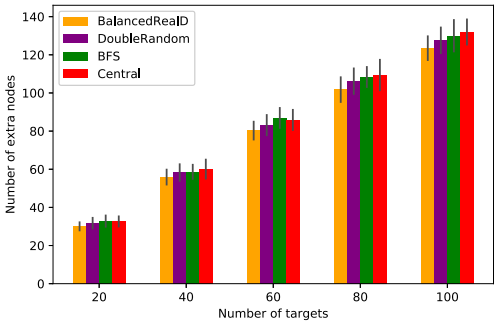

Fig. 34. Slashdot: Extra Nodes - Landmark.

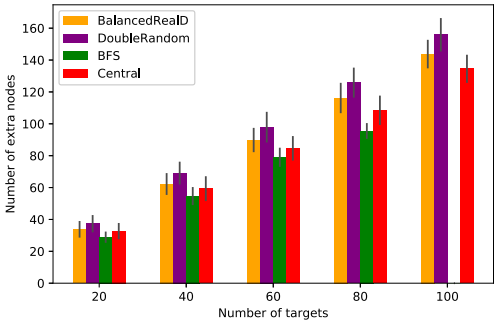

Fig. 35. Gowalla: Extra Nodes - Landmark.

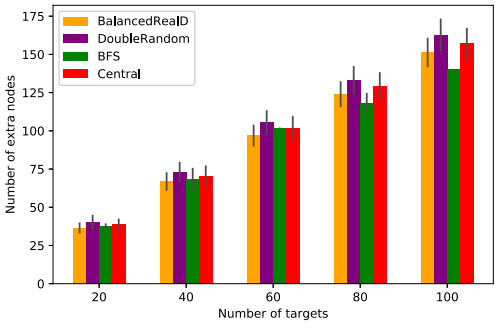

Fig. 36. Brightkite: Extra Nodes - Landmark.

sults of BFS only came from the successful online search. The performance of BalancedRealD is even better than the Central except for the Gowalla data set.

6. Discussion about target reachability

In our work, we assumed that all target nodes are reachable in the OSN, so the worst case in our online searching is to visit all nodes in the graph to achieve the connectivity of targets. Therefore, we preprocessed the real-world data sets by extracting the largest connected component to ensure the reachability of the targets before conducting online searching. However, in real-world OSNs, it could happen that the targets are not connected, although the chance may be slim, as nodes are connected well in online social networks [24]. Therefore, we hope to terminate the searching soon if it is highly likely that the targets are not reachable, so that we will not waste too much query resource.

From our experimental study, we realized that the number of queries issued in the online searching with BMS is about two times of the number of targets. Therefore, we set 10 times of the number of targets as the query threshold to terminate UMS. Actually, we can use BMS to relieve the assumption of graph connectivity. Specifically, after issuing a certain number of queries (as threshold), if the targets could not achieve connectivity, we can claim that the targets are not reachable or the cost to discover their connectivity is too high. Such threshold should be determined by the query cost a user could afford.

7. Related work

7.1. Search in social networks

Some attention has been paid to efficiently searching in a social network graph. [27] studied the searching on network graphs with power-law link distributions, containing a few nodes with very high degree and many with low degree. They proposed a number of local search strategies that utilize high degree nodes in power-law graphs. They also noticed that high connectivity nodes play the important role of hubs in communication and networking, which is exploited in designing efficient search algorithms. One of their proposed algorithms follows degree sequence. Specifically, at each step of searching, a neighboring node with a degree higher than the current node itself is selected for visiting so that a highest degree node will be reached quickly. Once a highest degree node is visited, an non-visited node of approximately second highest degree will be chosen. By following this degree sequence, one can reach a target node very quickly. In the paper [26], the same authors also addressed how to find a shortest path between a pair of nodes solely depending on local view on social networks. Their heuristic strategy also prioritizes high degree node in searching by virtue of the fact in our social community that if a person knows so many people, then he is more likely to know the target as well. Thus, nodes of high degrees are found important in searching social networks. Additionally, the authors in [31,32,39] discussed how to collect data from OSNs by crawling/sampling so as to analyze properties of social networks, such as topological properties.

Although our LMSD problem is also relevant to searching in social networks, it differs from the literature aforementioned. Specifically, rather than looking for a shortest path between a pair of nodes, we are more interested in the connectivity of a target group of nodes, usually more than two. Apparently, if we solely leverage on the high-degree nodes to find the shortest path for any pair of targeted nodes, the community including all the shortest paths will ensure the connectivity of the targeted nodes, thus LMSD problem being solved. However, the searching cost will be quite considerable. Therefore, we are motivated to design more efficient algorithms to solve the LMSD problem.

7.2. Subgraph connectivity

Our subgraph detection problem is relevant to the subgraph connectivity in the domain of graph mining. [14] and [15] proposes solutions for finding a subgraph that connects a set of query nodes in a graph, where the proximity between nodes is defined depending on the global topology of the graph. Specifically, they extracted subgraphs including nodes as close to the query nodes as possible, where the closeness is quantified by the similarity measure between two nodes. In its subsequent work, [16] redefined the proximity measures based on “cycle-free effective conductance” (cfec) and proposed some algorithms for optimizing the cfec measure. Another work [40] suggests the concept of *viewpoint neighborhood* analysis to identify neighbors of interest to a particular source in a dynamically evolving network, associating their measure with heat diffusion. [17] investigated the problem of connecting query nodes in a context-aware framework. They first employed modularity measure to partition the graph, and then studied the connectivity in both intra-community and inter-community levels. [18] proposed a random walk-based approach to find informative subgraphs associated with a group of query nodes in entity-relationship diagrams. Additionally, [19] addressed the searching for the densest community containing all query nodes with and without size constraint. Most recently, [41] examines the Steiner Maximum-Connected Subgraph (SMCS) problem: given a graph G and a set Q of query nodes, find the G’s induced subgraph that contains Q with the largest connectivity. Particularly, they addressed the minimal SMCS, which is the minimal subgraph of G with the maximum connectivity containing Q.

The main difference between our proposed problem and the above line of research is two-fold: (1) While the existing work addressed subgraph connectivity with pre-known global topology, thus from the perspective of social network “owner” (i.e., service provider), we instead consider the subgraph detection by a third-party analyst. (2) Unlike the traditional minimum subgraph detection problem the goal of which is to solely minimize the discovered subgraph which connects target nodes together, we are also concerned with the cost specified by the number of queries issued in OSNs for subgraph discovery, as many OSN web sites limit the number of web accesses per IP address per day to ensure the workload at OSN servers.

7.3. Local view based graph algorithms

Some researchers also noticed the importance of conducting graph mining or operation based on location information as often time the global information is not available. For example, [12] proposes local graph clustering methods to find a cluster of nodes by exploring a small region of the graph, which enable targeted clustering around a given seed node and are faster than traditional global graph clustering methods because their run time does not depend on the size of the input graph. Additionally, [11] proposes a local-search strategy, which searches in the neighborhood of a node to find the best community for the node. The difference between our work and the above work is that we consider a different search problem, and also, we particularly take advantage of topological properties of social networks in design of search strategies.

8. Conclusion

In this paper, we propose a problem of discovering a minimum subgraph covering a given group of nodes from the perspective of third-party analysts in OSNs, namely local-view based minimum subgraph detection (LMSD). Researching this problem has broad applications, for instance, finding a group of terrorists or malicious users in OSNs. To solve this problem, we propose two searching techniques, called Unbalanced Multiple-Subgraph (UMS) and Balanced Multiple-Subgraph (BMS), which are based on the well-known topological properties of social networks, including small-world phenomenon, power-law node degree distribution and the well-connectivity of nodes of high degree.

Through experiments over large-scale real-world and synthetic data sets, we evaluate the performance of our proposed techniques. The BMS technique performs better than UMS, which demonstrates that the well-connectivity property in social networks is not restricted to nodes of high degree in OSNs, rather, the entire OSNs are well connected, as any group of arbitrarily selected nodes can reach connectivity by a small number of node queries. Furthermore, the design principle in BMS of searching from subgraphs of low degree shows great impact on the efficiency in solving the LMSD problem. Our work sheds light on leveraging social network topological properties to conduct search efficiently, which may improve some of the existing searching-related research work in OSNs.

Declaration of competing Interest

All authors have participated in (a) conception and design, or analysis and interpretation of the data; (b) drafting the article or revising it critically for important intellectual content; and (c) approval of the final version.

This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue.

The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript

CRediT authorship contribution statement

Na Li: Conceptualization, Investigation, Methodology, Writing - original draft. **Sajal K. Das:** Supervision, Writing - review & editing.

Acknowledgment

This project is supported in part by the National Science Foundation (NSF) under grants DUE-1712496, CCF-1725755 and CCF-1533918. Any opinions, findings, and conclusions expressed in this paper are those of the authors, and do not necessarily reflect the views of NSF.

References

[1] N. Du, B. Wu, X. Pei, B. Wang, L. Xu, Community detection in large-scale social networks, in: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis, 2007, pp. 16–25.
[2] N.P. Nguyen, T.N. Dinh, Y. Xuan, M.T. Thai, Adaptive algorithms for detecting community structure in dynamic social networks, in: Proceedings of INFOCOM, 2011.
[3] P. Bedi, C. Sharma, Community detection in social networks, Wiley Interdisciplinary Reviews Data Mining and Knowledge Discovery (2016) 496–500.
[4] M. Wang, C. Wang, J.X. Yu, J. Zhang, Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework, VLDB 8 (10) (2015) 998–1009.
[5] L. Qin, J.X. Yu, R. Mao, Influential community search in large networks, VLDB Endowment 8 (5) (2015) 509–520.
[6] W. Fang, Graph pattern matching revised for social network analysis, in: Proceedings of the 15th International Conference on Database Theory, 2012, pp. 8–21.
[7] M. Papagelis, G. Das, N. Koudas, Sampling online social networks, IEEE Transactions on Knowledge and Data Engineering 25 (3) (2013) 662–676.
[8] A.D. Sarma, S. Gollapudi, M. Najork, R. Panigrapy, A sketch-based distance oracle for web-scale graphs, in: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining, 2010, p. 401410.
[9] X. Zhao, A. Sala, H. Zheng, B.Y. Zhao, Efficient shortest paths on massive social graphs, in: Proceedings of IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2011, p. 7786.
[10] C. Borgs, M. Brautbar, J. Chayes, S. Khanna, B. Lucier, The power of local information in social networks, Internet and Network Economics, Springer 7695 (2012) 406419.
[11] W. Cui, Y. Xiao, H. Wang, W. Wang, Local search of communities in large graphs, in: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, 2014, pp. 991–1002.
[12] H. Yin, A.R. Benson, J. Leskovec, D.F. Gleich, Local higher-order graph clustering, in: Proceedings of ACM SIGKDD, 2017, pp. 555–564.
[13] P. Basuchowdhuri, S. Sikdar, V. Nagarajan, K. Mishra, S. Gupta, S. Majumder, Fast detection of community structures using graph traversal in social networks, Knowledge and Information Systems (2018) 1–31.
[14] C. Faloutsos, K.S. McCurley, A. Tomkins, Fast discovery of connection subgraphs, in: Proceedings of ACM SIGKDD, 2004, pp. 118–127.
[15] H. Tong, C. Faloutsos, Center-piece subgraphs: problem definition and fast solutions, in: Proceedings of SIGKDD, 2006, pp. 404–413.
[16] Y. Koren, S.C. North, C. Volinsky, Measuring and extracting proximity graphs in networks, ACM Transactions on Knowledge Discovery from Data 1 (3) (2007).
[17] J. Cheng, Y. Ke, W. Ng, J.X. Yu, Context-aware object connection discovery in large graphs, in: Proceedings of IEEE ICDE, 2009, pp. 856–867.
[18] G. Kasneci, S. Elbassuoni, G. Weikum, Ming: mining informative entity relationship subgraphs, in: Proceedings of ACM conference on Information and knowledge management, 2009.
[19] M. Sozio, A. Gionis, The community-search problem and how to plan a successful cocktail party, in: Proceedings of ACM SIGKDD, 2010.
[20] C. Korte, S. Milgram, Acquaintance networks between racial groups: Application of the samll world method, Journal of Personality and Social Psychology 15 (2) (1970) 101–108.
[21] S. Milgram, The small world problem, Psychology Today 1 (61) (1967) 60–67.
[22] J. Travers, S. Milgram, An experimental study of the samll world problem, Sociometry 32 (425) (1969) 425–443.
[23] M.E.J. Newman, Assortative mixing in networks, Physical Review Letters 89 (20) (2002).
[24] C. Wilson, B. Boe, A. Sala, K.P. Puttaswamy, B.Y. Zhao, User interactions in social networks and their implications, in: Proceedings of EuroSys, 2009.
[25] M. Garey, D. Johnson, Computers and Intractability, Freeman, 1979.
[26] L. Adamic, E. Adar, How to search a social network, Social Networks 27 (3) (2005) 187–203.
[27] L.A. Adamic, R.M. Lukose, A.R. Puniyani, B.A. Huberman, Search in power-law networks, Physical Review 64 (4) (2001).
[28] A.-L. Barabasi, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[29] J. Leskovec, L. Backstrom, R. Kumar, A. Tomkins, Microscopic evolution of social networks, in: Proceedings of ACM SIGKDD, 2008, p. 462470.
[30] L. Kou, G. Markowsky, L. Berman, A fast algorithm for steiner trees, Acta Informatica 15 (2) (1981) 141–145.
[31] A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, B. Bhattacharjee, Measurement and analysis of online social networks, in: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement, 2007, p. 2942.
[32] S. Ye, J. Lang, F. Wu, Crawling online social graphs, in: Proceedings of 12th International AsiaPacific Web Conference, 2010, pp. 236–242.
[33] B. Viswanath, A. Mislove, M. Cha, K.P. Gummadi, On the evolution of user interaction in facebook, in: Proceedings of WOSN, 2009.
[34] J. Leskovec, Stanford Large Network Dataset Collection, 2019, (http://snap.stanford.edu/data/). [Online; accessed 14-March-2019].
[35] J. Leskovec, K. Lang, A. Dasgupta, M. Mahoney, Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters, Internet Mathematics 6 (1) (2009) 29–123.
[36] E. Cho, S.A. Myers, J. Leskovec, Friendship and mobility: Friendship and mobility: User movement in location-based social networks, in: Proceedings of ACM SIGKDD, 2011, pp. 1082–1090.
[37] A. Clauset, C.R. Shalizi, M.E.J. Newman, Power-law distributions in empirical data, Society for Industrial and Applied Mathematics 51 (4) (2009) 661–703.
[38] J. Alstott, E. Bullmore, D. Plenz, powerlaw: a python package for analysis of heavy-tailed distributions, PLoS ONE 9 (1) (2014) e95816.
[39] M. Gjoka, M. Kurant, C.T. Butts, A. Markopoulou, Practical recommendations on crawling online social networks, IEEE Journal on Selected Areas in Communications 29 (9) (2011) 1872–1892.
[40] S. Asur, S. Parthasarathy, A viewpoint-based approach for interaction graph analysis, in: Proceedings of ACM SIGKDD, 2009, pp. 79–88.
[41] J. Hu, X. Wu, R. Cheng, S. Luo, Y. Fang, On minimal steiner maximum-connected subgraph queries, IEEE Transactions on Knowledge and Data Engineering 29 (11) (2017) 2455–2469.