

Spatial-Aware Local Community Detection Guided by Dominance Relation

Li Ni, Hefei Xu, Yiwen Zhang and Wenjian Luo

Abstract—The problem of finding the spatial-aware community for a given node has been defined and investigated in geosocial networks. However, existing studies suffer from two limitations: a) the criteria of defining communities are determined by parameters, which are difficult to set; b) algorithms may require global information and are not suitable for situations where the network is incomplete. Therefore, we propose spatial-aware local community detection (SLCD), which finds the spatial-aware local community with only local information and defines the community based on the difference in the sparseness of edges inside and outside the community. Specifically, to address the SLCD problem, we design a novel spatial aware local community detection algorithm based on dominance relation, but this algorithm incurs high cost. To further improve the efficiency, we propose an approximate algorithm. Experimental results demonstrate that the proposed approximate algorithm outperforms the comparison algorithms.

Index Terms—Geo-social networks; Local community detection; Spatial-aware local community detection; Dominance relation

I. INTRODUCTION

With the increasing popularity of location-based services, geosocial networks have emerged [1]. Geosocial networks contain users' social relations and geographic location information. In geosocial networks, one of the most critical tasks is detecting spatial-aware communities [2], which has broad application prospects in many location-based social services, such as event recommendation, social marketing, and geosocial data analysis [3].

In this paper, we study the problem of spatial-aware local community detection (SLCD) in geosocial networks. Specifically, for a geosocial network and a given node, the objective is to find the spatial-aware local community to which the given node belongs. The following two properties hold: 1) only local information is used in the process of detecting the community; and 2) the community satisfies both structural and spatial cohesiveness. Structural cohesiveness means that nodes inside the community are relatively tightly connected,

and nodes inside and outside the community are relatively sparsely connected, while spatial cohesiveness means that the locations of nodes in the same community are close to each other.

Prior Work. The studies on finding communities contain global community detection [4], [5], local community detection [6], [7] and community search [8], [9]. Global community detection algorithms aim to detect all communities in social networks [4], [10], [11]. Most global community detection studies only use topology information to detect communities [5], [12]. In real life, users' spatial location can affect social relationships because offline social activities are constrained by geography [13]–[15]. Therefore, some work has considered the user's location information [14]–[17]. Global community detection methods often require global information of the network, such as the total number of edges [18]. However, because of trade secrets, global information about entire networks may be unavailable or expensive to obtain. In addition, when users want to know the local community to which the given node belongs, it is not necessary to mine all the communities in the network [18].

To compensate for these shortcomings, local community detection has been investigated, which can quickly detect the community that contains the given node with only local information [7], [19]–[21]. Similar to the local community detection problem, community search aims to find a subgraph containing a set of given nodes. Some community search works need global information of social networks [8], and some do not [22], [23]. These above studies consider only the link between nodes [7], [8], [19], [20], [22], ignoring the nodes' locations, so the detected communities may not be spatially cohesive and may not be suitable for some location-based services [3], [15], [24].

To obtain a community that satisfies both structural and spatial cohesiveness, spatial-aware community search has attracted attention [1], [3]. Existing studies adopt different spatial constraints to restrict the geographic location of nodes to ensure spatial cohesiveness and place a minimum degree constraint on nodes to guarantee structural cohesiveness [1]–[3], [25]. For example, Fang et al. regarded a k -core structure with the minimum coverage circle (MCC) as a spatial-aware community [26], which considers the location information of nodes and could obtain a spatially cohesive community.

However, existing studies need to set parameters such as parameter k in the k -core, which is not easy for users to set [27]. If k is set to be large, the k -core structure does not exist. On the other hand, if k is set to be small, the k -core structure may not be structurally cohesive. Taking Eve in Fig. 1 as

This work was supported by the National Key Research and Development Plan of China [No.2019YFB1704101], and the National Natural Science Foundation of China [No.62106004, No.U1936220 and No.61872002] (*Corresponding author: Yiwen Zhang.*)

Li Ni, Hefei Xu, and Yiwen Zhang are with the School of Computer Science and Technology, Anhui University, Hefei, Anhui 230601, China. (e-mail: nlcs@mail.ustc.edu.cn, e20301226@stu.ahu.edu.cn, zhangyiwen@ahu.edu.cn).

Wenjian Luo is with School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China. (e-mail: luowenjian@hit.edu.cn).

Note: This work has been submitted to the IEEE for possible publication. Copyright may be transferred without notice, after which this version may no longer be accessible.

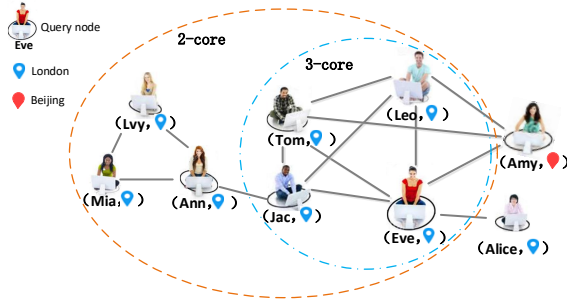


Fig. 1: Toy geosocial network and k -core structure detected by the *Exact* algorithm [3]

an example, when $k \geq 4$, the community that contains Eve does not exist; when $k = 2$ or 1 , the k -core structure is not sufficiently cohesive. When $k = 3$, the 3-core containing Eve is suitable, but the 3-core containing Ann does not exist. In addition, some methods require global information [26] and are not suitable for incomplete networks. For situations in which global information about the network is unavailable or the user is only interested in the community to which the given node belongs, we propose SLCD, which uses only local information to detect the community. To avoid setting the parameter, we adopt the difference between inside and outside the community to detect communities, and propose a parameter-free Spatial-aware Local community detection method based on Dominance Relation (SLDR). The SLDR algorithm can obtain the community of spatial cohesiveness and structural cohesiveness. However, obtaining the derived communities by the community derivation process incurs a high cost. To address this bottleneck, we propose an approximation community derivation process. The basic idea is to reduce the number of derived communities.

The contributions of this work are summarized as follows:

- We propose the SLCD problem and define dominance relation between communities based on structural and spatial cohesiveness of the communities.
- We propose the SLDR algorithm without parameters, which iteratively performs community derivation and community filtration. We further propose the approximate SLDR algorithm with the approximate community derivation process.
- We evaluate the effectiveness of the proposed algorithm on synthetic and real datasets. The experimental results show that the approximation algorithm outperforms other comparison algorithms.

The rest of the paper is organized as follows. Section II reviews the related works. Section III presents the preliminaries, including the local modularity and dominance relation. Section IV first introduces the SLCD problem and then designs the SLDR algorithm and its approximation algorithm. Section V conducts experiments on synthetic and real geosocial networks. Section VI concludes our work.

II. RELATED WORK

Studies on finding communities contain community detection [4] [7] and community search [22]. Community detection

can be divided into global community detection and local community detection [28].

A. Global Community Detection

Community detection aims to detect all communities in social networks [5], [29], [30]. Most community detection works consider only link information when detecting the community [31], [32]. Newman et al. [4] defined modularity and designed the greedy algorithm based on modularity to partition communities in the network. Pizzuti [33] designed a multi-objective evolutionary algorithm for community detection. It divides the nodes into different communities by optimizing the tightness of intral-connections inside each community and the sparsity of inter-connections between different communities. Li et al. [34] propose a method based on network representation learning which combines node embedding and community embedding to detect communities in social networks.

Some works on geosocial networks consider the effect of users' spatial information, aiming at finding communities that satisfy spatial and structural cohesiveness [15], such as the reports by [16], [17]. The works in [15] [16] used the location information of the node to weight the edges to transform the unweighted network into a weighted network and then detect the communities. Zhang et al [17] adopted k -core and the pairwise similarity between users based on attribute values (e.g., users' geo-locations) to guarantee the cohesiveness of a community from both structural and vertex attributes. All of these works detect all communities from a global view rather than a local view, which is different from our work.

B. Local Community Detection and Community Search

The goal of local community detection is to obtain the local community that contains the given node with only local information [19], [20]. Researchers have proposed various approaches for local community detection [18], [35]–[38]. Luo et al. [7] proposed a local modularity LQ and designed modularity optimization algorithms based on LQ . He et al. [28] developed a community detection method based on the local spectral subspace, which is defined based on the Krylov subspace. Lyu et al. [37] proposed an EA-based method for local community detection with some effective strategies in terms of individual representation, fitness evaluation, the local search operation, etc. Most studies on local community detection do not consider the location information of nodes and are not suitable for detecting communities in geosocial networks.

Another related line of work on finding the community that contains the query node is community search. For a graph and a set of query nodes, the task of community search is finding a connected subgraph based on query parameters [27], [39]. Huang et al. [40] proposed a k -truss community model and designed a tree index to detect local communities in the network efficiently. Liu et al. [41] studied the SCkT search problem, which aims to find a triangle-connected k -truss containing query nodes with sizes no larger than a given threshold as a community. They investigated *expansion* and *shrinking* strategies to detect communities in networks. These studies

on community search employ only link information to detect communities without considering the location information of nodes, which is not suitable for detecting communities that satisfy both structural and spatial cohesiveness.

Local community detection is similar to community search; both detect the community that contains the given/query node. The two are not identical, however. The difference between the two includes the definition of the community and whether global information is used when detecting the community.

C. Spatial-aware Community Search

Most existing studies employ the k -core model [1]–[3], [26], [42] to ensure the structural cohesiveness of communities. These studies use different spatial constraints, such as the minimum MCC constraint, range constraint, and k -nearest neighbor constraint, to ensure the spatial cohesiveness of communities [2], [26]. For example, Fang et al. [3], [26] proposed the exact solution to detect the k -core structure covered by the smallest MCC as a community. Wang et al. [1] ensured that the nodes in the same community are geographically close by a radius-bounded circle. However, the value of k in these methods [1], [26], [42] is hard to set. When k is set larger than the degree of the query node, the community does not exist. Even if k is less than the degree of the query node, the community may not be found because the query node and its neighbors may not satisfy the minimum degree constraint; on the other hand, when k is set to be small, too many nodes satisfy the minimum degree constraint, thus making the detected community possibly not structurally cohesive enough.

To our knowledge, existing studies on finding spatial-aware communities containing given nodes focus mainly on spatial-aware community search (SAC). SLCD has not received much attention. The goal of SLCD is to detect the local community that satisfies both structural and spatial cohesiveness only with local information. The differences between SAC and SLCD are evident in the following two aspects.

- For SAC, the criteria of the definition of structural cohesiveness are based on query parameters (e.g., k -core [22]). In contrast, the criteria of defining structural cohesiveness for SLCD usually take advantage of the difference in the sparseness of edges inside and outside the community (e.g., local modularity).
- The SLCD algorithms use only local information when detecting communities, while SAC algorithms have no restriction on whether to use global information. That is, some algorithms require global information [40], and some do not [43].

III. PRELIMINARIES

In this section, we first introduce the local modularity and then introduce some relevant definitions about dominance relation.

A. Local Modularity

Luo et al. [20] proposed the local modularity called M to evaluate the quality of the community. Local modularity M is

based on internal and external edges of the community, defined as follows.

$$M = \frac{e_I}{e_O}, \quad (1)$$

where e_I is the number of internal edges of the community and e_O is the number of external edges of the community. If the number of internal edges is larger and the number of external edges is smaller, the quality of the community is better.

B. Dominance Relation

Given the objective function space F and the solution space X , some relevant definitions about dominance relation are as follows.

Definition 1. (Dominance relation [44]). Given maximization objective functions: $f_1(x), f_2(x), f_3(x), \dots, f_n(x) \in F$, two solutions: $x_1, x_2 \in X$, $\forall i \in \{1, 2, \dots, n\}, \exists j \in \{1, 2, \dots, n\}$, if $f_i(x_1) \leq f_i(x_2)$ and $f_j(x_1) < f_j(x_2)$, then solution x_2 dominates solution x_1 or solution x_1 is dominated by solution x_2 , denoted as $x_1 < x_2$.

Definition 2. (Nondominated solution and dominated solution [44]). Among solution space X , solution $x \in X$ is a nondominated solution or Pareto-optimal solution if it cannot be dominated by any solution in X . Otherwise, x is a dominated solution.

Dominance relation is often used in multi-objective optimization to find nondominated solution [45]. To obtain a set of nondominated solutions, Liu et al. proposed the BNSA algorithm [46]. For the set of multiple solutions $\{x_1, x_2, x_3, \dots, x_n\}$ and two maximization objective functions f_1 and f_2 , the BNSA algorithm [46] first sorts solutions in $\{x_1, x_2, x_3, \dots, x_n\}$ in descending order of the f_1 value. If the two solutions have the same value of f_1 , the solutions are sorted in descending order of f_2 value. Then, starting from the second solution, each solution x_i is processed as follows. If it is dominated by the previous solution, x_i is the dominated solution, which is deleted from sorted solutions. Otherwise, x_i is a nondominated solution, which is retained in sorted solutions. The first solution is a nondominated solution because it has the maximum value of f_1 , which implies that no other solution can dominate it. After all solutions except the first solution are processed, we can obtain nondominated solutions. Supposing that n is the size of the solution set, the time complexity of the sorting step and comparison step are $O(n \log n)$ and $O(n)$, respectively. Therefore, the time complexity of the BNSA algorithm is $O(n \log n)$ [46].

IV. PROPOSED ALGORITHM

A. Problem Statement and Community Dominance Relation

We start this subsection with an introduction to geosocial networks. Then we introduce the SLCD problem and community dominance relation.

A geosocial network is a graph with node location information. Let $G(V, E)$ represent a geosocial network, where E represents the edge set and V represents the node set. Each node in V has location information, which is often represented by horizontal and vertical coordinates.

Problem 1. (SLCD Problem) For a geosocial network and a given node, SLCD aims to find the spatial-aware local community, satisfying the following properties:

- **Connectivity.** The community containing the given node and nodes in the community are directly or indirectly connected.
- **Structural cohesiveness.** The nodes inside the community are relatively tightly connected to each other, while nodes inside and outside the community are relatively sparsely connected.
- **Spatial cohesiveness.** The locations of nodes in the same community are close to each other.
- **Only local information.** Only local information is used when detecting the community. For example, only nodes and edges in or near the community are accessed.

The difference between SLCD and SAC problems [3] is mainly in two aspects: 1) One is structural cohesiveness. The criteria of structural cohesiveness of the SLCD problem are based on the difference in the sparseness of edges inside and outside the community. The criteria of structural cohesiveness of the SAC problem are based on query parameters (e.g., k -core). 2) The other is that the SLCD problem uses only local information, while the SAC problem has no restrictions on global information of the network. For better understanding, here, we take the *Exact* algorithm [3] and the proposed method (Section IV-B) as examples. The first step of the *Exact* algorithm is to traverse all the nodes in the network to extract the k -core subgraph. All nodes in geo-networks are accessed, so the *Exact* algorithm uses global information. Our proposed method only accesses the nodes in or near the detected community. That is, only local information is utilized.

Here, we model the SLCD problem with two objective. The first objective is to maximize the structural cohesiveness of the community. The other is to maximize the spatial cohesiveness of the community. Specifically, we use local community M (calculated by (1)) to measure structural cohesiveness, while S is adopted for measuring spatial cohesiveness, formulated as

$$\begin{cases} \max f_1 = M = \frac{e_I}{e_O} \\ \max f_2 = S = -\frac{\sum_{i,j \in C} d(i,j)}{|C| * (|C|-1)}, \end{cases} \quad (2)$$

where $|C|$ denotes the size of community C and $d(i, j)$ denotes the distance between nodes i and j . S is a variant of average distance between nodes within the community, which is used to measure the degree of community spatial cohesiveness in [3], [26]. Here, we use S as the optimization objective to improve the community spatial cohesiveness. Maximizing M and S of the community could make the links within a community dense while the locations of nodes in the same community are close.

When detecting communities in geosocial networks, maximizing the first objective may make the second objective worse, and vice versa. Specifically, maximizing the first objective adds nodes that are more structurally connected to the community into the community. If the node is far away from the community, the second objective will worsen. For example, for the community {Led, Tom, Jac, Eve} in Fig.

TABLE I: Meanings of some notations

Notation	Meaning
G	geo-social network
C	local community C
N_i	set of neighbor nodes of node i
N_C	set of neighbor nodes community C
M_C	M of community C
S_C	S of community C
ND	set of nondominated communities
NDE	set of nondominated communities that have not been expanded
HND	set of nondominated communities that have been expanded
D	set of derived communities that are expanded from nondominated communities

1 (or the community that contains the two closest nodes in geosocial networks), maximizing the first objective adds Eve (one node) to the community, making the second objective worse. Maximizing the second objective adds some nodes that are close to the community to the community. If these nodes are structurally sparsely connected to the community, the first objective will worsen. In summary, the first objective and the second objective are potentially conflicting.

On the basis of the M value and S value of the community, we define the dominance relation between communities, nondominated community, dominated community, and derived community.

Definition 3. (Community dominance relation). Given community C_1 and community C_2 , we say that community C_2 is dominated by community C_1 or C_1 dominates C_2 (denoted as $C_2 < C_1$) if $M_{C_1} \geq M_{C_2}$ and $S_{C_1} > S_{C_2}$, or $S_{C_1} \geq S_{C_2}$ and $M_{C_1} > M_{C_2}$, where M_C (S_C) is M (S) of the community C .

Definition 4. (Nondominated community and dominated community). Among a given set of communities, community C is a nondominated community if it cannot be dominated by any communities. Otherwise, C is a dominated community.

Definition 5. (Derived community). Given a nondominated community C and its neighbor nodes set N_C , the community (e.g., $C \cup \{v\}$) expanded by adding one node v in N_C to community C is called the derived community.

B. Basic Algorithm

To address the SLCD problem, we design a novel Spatial-aware Local community detection algorithm with Dominance Relation (SLDR). We first introduce the SLDR algorithm and then provide a detailed description of its two key processes.

Table I show some notations used in this paper.

1) *SLDR Algorithm:* Based on the definitions described in section IV-A, we propose the SLDR algorithm for the SLCD problem, as shown in Algorithm 1. The basic idea of this algorithm is to maximize the S and M of the community by iteratively performing community derivation (section IV-B2)

Algorithm 1 SLDR

Input: G, v
Output: community that v belongs to

```

1:  $C \leftarrow \{v\}$ 
2:  $N_C \leftarrow N_v$ 
3:  $M_C \leftarrow 0, S_C \leftarrow -\infty$ 
4:  $HND \leftarrow \emptyset$ 
5:  $ND \leftarrow \{C\}$ 
6:  $NDE \leftarrow \{C\}$ 
7: while  $NDE \neq \emptyset$  do
8:    $D \leftarrow$  Community Derivation ( $NDE$ ) (Alg. 2)
9:    $ND \leftarrow$  Community Filtration ( $D \cup ND$ ) (Alg. 3)
10:   $HND \leftarrow (ND \cap HND) \cup (NDE \cap ND)$ 
11:   $NDE \leftarrow ND - HND$ 
12: end while
13: Select one community  $C$  from  $ND$ 

```

and community filtration (section IV-B3). The community derivation process aims to expand the community by generating communities derived from nondominated communities. The community filtration process removes the dominated communities in derived communities to obtain the nondominated communities.

The general process of the SLDR algorithm is as follows. Initially, community C contains v and NDE contains community C (lines 1–6). First, Algorithm 1 expands the communities in NDE to obtain derived communities D by the community derivation process (section IV-B2). Some nondominated communities in ND may be dominated by communities in D and become dominated communities. Therefore, the algorithm removes the dominated communities from $D \cup ND$ to obtain nondominated communities ND by the community filtration process (section IV-B3). Then, the algorithm updates HND (line 10), including removing the dominated community in HND (e.g., $ND \cap HND$) and adding the newly expanded nondominated community (e.g., $NDE \cap ND$) to HND . Next, the algorithm obtains the nondominated community set NDE by deleting the processed nondominated communities from ND (line 11). If NDE is not empty, the algorithm continues the process of derivation and filtration. By iteratively performing community derivation and community filtration, the local communities are continuously expanded and optimized. Otherwise, the algorithm jumps out of the loop because the communities will not be optimized further at this point. Finally, we select one community from ND as the final community (line 13). ND contains multiple communities, which have been sorted in descending order by the value of M . In order that the M and S values of the selected community are balanced, the community in the middle of ND is selected. That is, the $\lfloor |ND|/2 \rfloor$ th community in ND is selected where $|ND|$ is the number of communities in ND .

Example 1. We use node a as the given node and the geosocial network in Fig. 2 to illustrate the process of the SLDR algorithm. The states of D , ND , HND and NDE after the first and second loops are shown in Fig. 3. Initially, the community is $\{a\}$, and NDE contains $\{a\}$. Since NDE is

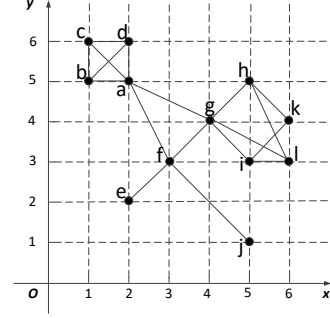


Fig. 2: Example of a geosocial network. Each node has location (x, y) in a two-dimensional space

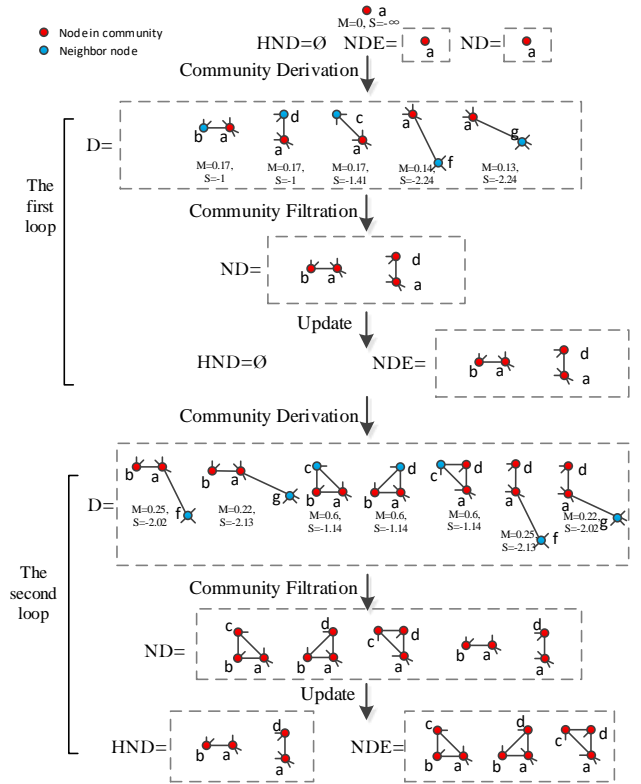


Fig. 3: Illustration of the SLDR algorithm

not empty, the algorithm enters the first loop. After derived community set D is obtained, the communities in $D \cup HND$ are screened to obtain nondominated communities $ND = \{\{a, b\}, \{a, d\}\}$. Then, HND is empty, and $NDE \cap ND = \{\{a\}\} \cap \{\{a, b\}, \{a, d\}\} = \emptyset$, so HND is empty. Since community set $NDE = ND - HND = \{\{a, b\}, \{a, d\}\}$ is not empty, the algorithm enters the next loop. The second loop performs community derivation and community filtration to obtain D and ND , respectively. Since $(ND \cap HND)$ is empty and $NDE \cap ND = \{\{a, b\}, \{a, d\}\}$, HND is $\{\{a, b\}, \{a, d\}\}$. Communities in HND has been processed before and do not be added to the NDE . The community set $NDE = \{\{a, b, c\}, \{a, b, d\}, \{a, c, d\}\}$. Since NDE is not empty, the algorithm continues community derivation and community filtration processes, omitted to save space.

Algorithm 2 Community Derivation

Input: NDE
Output: D

- 1: $D \leftarrow \emptyset$
- 2: **for** each community C in NDE **do**
- 3: **for** each node u in N_C **do**
- 4: $C' = C \cup \{u\}$
- 5: **if** C' is in D **then**
- 6: continue
- 7: **else**
- 8: compute $M_{C'}$ and $S_{C'}$
- 9: $N_{C'} = N_C \cup N_u - C'$
- 10: add C' to D
- 11: **end if**
- 12: **end for**
- 13: **end for**
- 14: **return** D

Although the geosocial network G is the input of the SLDR algorithm, the SLDR algorithm visits only the neighbor nodes of the communities to obtain derived communities. Therefore, the proposed algorithm only uses the local information instead of the global information of the geosocial network.

2) *Community Derivation Process:* The community derivation process aims to expand the community by generating derived communities of nondominated communities in NDE . The essential operation of obtaining a derived community is to add one neighbor node in N_C to community C to form a newly derived community C' .

Algorithm 2 shows the process of community derivation, which is described as follows. First, derived community set D is empty (line 1). Then, for each community C in NDE and for each node u in N_C , the following steps are performed (lines 4-10): a) The algorithm obtains the derived community C' by adding node u to C ; b) If C' is not in D , the algorithm calculates $M_{C'}$ and $S_{C'}$, obtains $N_{C'}$, and adds the derived community C' to D . Finally, the derived community set D is obtained.

Example 2. We continue with example 1. In the first loop, initially, $NDE = \{\{a\}\}$ and $D = \emptyset$. For node b in $N_{\{a\}}$ where $N_{\{a\}} = \{b, c, d, f, g\}$, the following steps are performed: i) The algorithm obtains a derived community $C' = \{a, b\}$; ii) Since $\{a, b\}$ is not in D , the algorithm computes $M_{\{a,b\}}$ and $S_{\{a,b\}}$ to be 0.17 and -1, respectively; iii) The algorithm obtains $N_{\{a,b\}} = \{c, d, f, g\}$ and adds the derived community $\{a, b\}$ to D . Similarly, c, d, f and g are added to community C . Therefore, $D = \{\{a, b\}, \{a, d\}, \{a, c\}, \{a, f\}, \{a, g\}\}$.

3) *Community Filtration Process:* In the community filtering process, we apply BNSA [46] (Section III-B) to obtain nondominated communities by removing dominated communities from derived communities.

Algorithm 3 shows the process of community filtration. First, the algorithm sorts the communities in $D \cup HND$ in descending order of M and S values to obtain the sorted list ND (line 1). If the M values of two communities are equal, the communities are sorted in descending order of S

Algorithm 3 Community Filtration

Input: $D \cup ND$
Output: ND

- 1: $ND \leftarrow$ sort communities in $D \cup ND$
- 2: **for** each community C in ND **do**
- 3: **if** $C < C^{prior}$ **then**
- 4: remove C from ND
- 5: **else**
- 6: continue
- 7: **end if**
- 8: **end for**
- 9: **return** ND

value. For convenience, for community C , let C^{prior} represent the previous community of C . Then, the algorithm removes dominated communities (lines 2-6). Specifically, starting with the second community in ND , each community is processed as follows: If community C is dominated by C^{prior} , then community C is a dominated community, so the algorithm removes C from ND ; otherwise, community C is a nondominated community, which is retained in ND . The first community in ND is a nondominated community because it has the largest M among all communities in ND . Finally, Algorithm 3 returns the nondominated communities in ND .

Example 3. Continue with the example 2. In the first loop, sort communities in $D \cup HND$ to obtain $ND = \{\{a, b\}, \{a, d\}, \{a, c\}, \{a, f\}, \{a, g\}\}$. The first community $\{a, b\}$ in ND is a nondominated community. For the second community $\{a, d\}$, since $M_{\{a,d\}}$ and $S_{\{a,d\}}$ are equal to $M_{\{a,b\}}$ and $S_{\{a,b\}}$, respectively, $\{a, d\}$ is not dominated by $\{a, b\}$. Therefore, $\{a, d\}$ is a nondominated community. For the third community $\{a, c\}$, since $S_{\{a,c\}}$ is less than $S_{\{a,d\}}$, $\{a, c\}$ is not a nondominated community. Community $\{a, c\}$ is removed. Similarly, $\{a, f\}$ and $\{a, g\}$ are dominated communities, which are removed from ND .

C. Approximate Algorithm

In this section, we first analyze the SLDR algorithm. Then, we introduce an approximate community derivation process. In addition, the algorithm that uses the approximate community derivation process is the proposed approximation algorithm, called AppSLDR algorithm.

In the community derivation process, for a nondominated community C , each neighbor node in N_C is added to community C to obtain a derived community. The number of derived communities of a community is equal to the number of neighbor nodes of this community. For each derived community C' , the community filtration process needs to compute $M_{C'}$ and $S_{C'}$. If there are too many nondominated communities or too many neighbor nodes of a nondominated community, community derivation and community filtration processes have high computational costs. Thus, we developed a more efficient approximate community derivation process.

Observation. We start with an important phenomenon. During the execution of the SLDR algorithm, we observe a phenomenon: in most cases, the number of derived communities

is much larger than that of nondominated communities. This phenomenon shows that many of the communities obtained in the process of community derivation are dominated communities. This means that the community derivation process spends considerable time computing those communities that will be eliminated in the community filtration process.

Based on the above observation, we developed an efficient approximation community derivation process. The basic idea is that we use part of the neighbor nodes of the nondominated community to obtain derived communities rather than all neighbor nodes. The M value of the derived community, obtained by the node with few internal edges and many external edges, may be relatively small, which makes the derived community have a high probability of being a dominated community. These nodes are no longer combined with the community to generate derived communities, thereby reducing time.

The method of selecting nodes is as follows: We first sort the nodes in N_C in descending order of the inward ratio [28], which is defined as the ratio of inward edges to the out-degree. Naturally, based on the value of inward ratio, the nodes in N_C are evenly divided into upper, middle and lower levels. The nodes of the upper level usually generate higher quality communities. So, we choose the top $\lceil |N_C|/3 \rceil$ nodes in the sorted list, termed SN_C . This method can be implemented by replacing N_C with SN_C in line 3 and replacing N_C with SN_C in line 9 of Algorithm 2, where SN_C represents the top $\lceil |N_C|/3 \rceil$ nodes in the sorted neighbor nodes of community C' .

Example 4. Fig. 4 shows the process of approximate community derivation. In the first loop, Fig. 4 shows two derived communities $\{a, b\}$ and $\{a, c\}$, while five derived communities are generated in Fig. 3. Similarly, for the second loop, two derived communities are shown in Fig. 4, while seven derived communities are generated in Fig. 3. Although the number of derived communities generated by the approximation process is less than that of the original community derivation, in the second loop, NDE in Fig. 3 is similar to NDE in Fig. 4, which ensures that the performance of the approximation algorithm is close to the basic algorithm.

V. EXPERIMENTS

In this section, we conduct comprehensive experiments to test the proposed algorithms. We first introduce the experimental settings, including datasets, evaluation metrics and comparison algorithms. Implementation of this work was carried out using Centos7 (CPU: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40 GHz, memory: 200 GB). The algorithms were implemented using Python 3.7 programming language.

A. Experimental Settings

1) *Datasets*: We tested our algorithm on four real and two synthetic datasets. The statistics of these datasets are summarized in Table II. For each dataset, we evenly select 200 given nodes for the experiments. Each node is selected as the given node for spatial-aware local community detection,

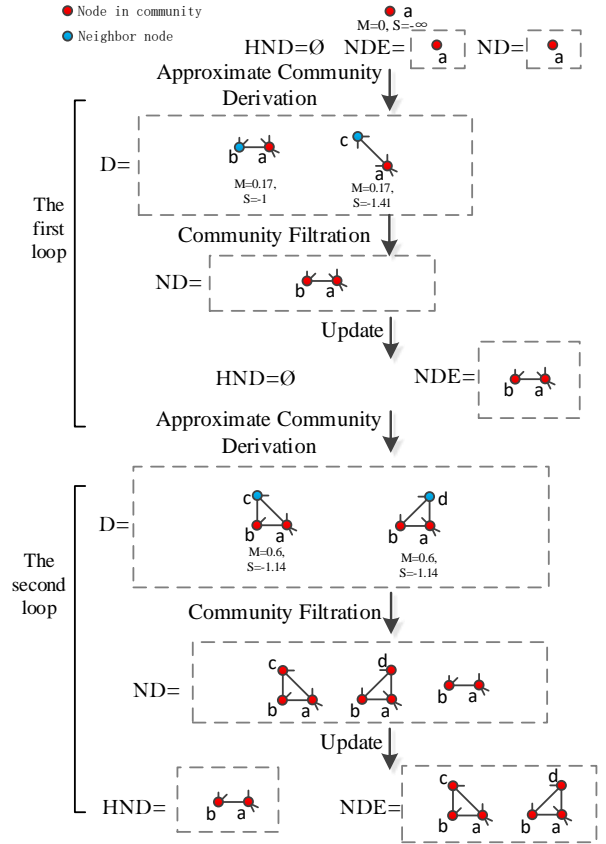


Fig. 4: Illustration of the AppSLDR algorithm

and then the average values of metrics are calculated for all selected nodes. The four real datasets are *Brightkite*¹, *Gowalla*¹, *Flickr*², and *Foursquare*³. In the above four real datasets, each node is a user, and each edge is the friendship between two users. We implement the processing of these datasets referring to [26], introduced as follows: a) *Brightkite* contains 4,491,143 checkin records collected from 772,783 different places from April 2008 to October 2010. The user's geographic location is the place that the user checks most frequently. b) *Gowalla* contains 6,442,892 checkin records collected from 1280,969 places. Similar to the *Brightkite* dataset, the place that the user checks most often is marked as the user's geographic location. c) *Flickr* contains locations where photos were taken. The location where a user took photos most frequently is marked as the user's geographic location. d) *Foursquare* contains 33,278,683 checkin records, obtained by crawling the Foursquare website. Each user in the *Foursquare* dataset has a location of he/her hometown position, which is regarded as the user's geographic location.

We also conducted experiments on synthetic datasets. Synthetic networks are generated by a graph generator named GTGraph⁴, following the method in [1], [26]. We obtained the synthetic datasets by the following two steps: (1) Generate a

¹<http://snap.stanford.edu/data/index.html>

²<https://www.flickr.com/>

³https://archive.org/details/201309_foursquare_dataset_umn

⁴<http://www.cse.psu.edu/~madduri/software/GTgraph/>

TABLE II: Statistics of datasets

Type	Datasets	#Vertices	#Edges	Average Degree
Real	<i>Brightkite</i>	51406	197167	7.67
	<i>Gowalla</i>	107092	456830	8.53
	<i>Flickr</i>	214698	2096306	19.5
	<i>Foursquare</i>	2127093	8460352	8.12
Synthetic	<i>Syn1</i>	5000	20000	8
	<i>Syn2</i>	200000	800000	8

social network without node location information by the R-MAT graph generator in GTGraph. The degrees of the nodes in the generated network obey a power-law distribution, and the default parameter values of the GTGraph are adopted; (2) Generate location information for all nodes in the social network. We randomly generate a location coordinate for each node with a location range of $[0, 1] \times [0, 1]$. This process is repeated for each node until all nodes have location coordinates. Based on these steps, we generated two synthetic datasets: *Syn1* and *Syn2*.

2) *Evaluation metrics*: When measuring the quality of the community, we take both structural and spatial cohesiveness of the community into account. The metric *communitude* [47], [48] is adopted to measure the degree of structural cohesiveness of the community; d_{avg} and d_{IO} are adopted to measure the degree of spatial cohesiveness of the community. These three metrics are introduced as follows:

a) The metric *communitude* [47], [48] measures the structural cohesiveness based on internal and external structural differences of communities, and it is calculated as follows:

$$communitude(C) = \frac{\frac{e[C]}{m} - (\frac{D[C]}{2m})^2}{\sqrt{(\frac{D[C]}{2m})^2(1 - (\frac{D[C]}{2m})^2)}}, \quad (3)$$

where m is the total number of edges in the network, $e[C]$ is the number of internal edges of C , and $D[C]$ is the sum of degrees of the nodes in C . A higher value of *communitude* indicates better structural cohesiveness of the community.

b) Here, d_{avg} measures the spatial proximity of nodes in the community. It is defined as the average distance between node pairs in C . The smaller the value of d_{avg} is, the better the spatial cohesiveness of the community.

c) In addition, d_{IO} measures spatial cohesiveness based on the geographical proximity of internal nodes and external nodes of the community, which is calculated by (4).

$$d_{IO} = \frac{d_{avg}}{\sum_{i \in C} \sum_{j \in N_C} d(i, j) / (|C| * |N_C|)}, \quad (4)$$

where $d(i, j)$ is the distance between node i and node j . The smaller the value of d_{IO} is, the better the spatial cohesiveness of the community.

3) *Comparison algorithms*: We compare the proposed method with a local community detection method (i.e., M method [20]), a global spatial-aware community detection method (i.e., Geomod [15]), and a spatial-aware community search method (i.e., AppAcc [26]). We briefly describe these comparison algorithms:

- M method [20]. This method starts with a given node and then finds a community with the largest local modularity M .
- Geomod [15]. Geomod is a global spatial-aware community detection algorithm that detects all communities in geosocial networks. In our experiments, we select the community containing the given node from all detected communities. In addition, parameter n is set to 2.
- AppAcc [3]. The parameters of the AppAcc algorithm follow the experimental setting in [3]. Specifically, k is set to 4, and ϵ_A is set to 0.5. Since the exact algorithm (*Exact+*) in [3] is slow on large data sets, we take the AppAcc algorithm (the most accurate approximation algorithm [3]) as the comparison algorithm.

Since the SLDR algorithm runs slowly, we use the AppSLDR algorithm instead of the SLDR algorithm to compare with other algorithms. In addition, if the running time of the AppSLDR algorithm for a node is longer than two hours, we terminate it and select one community from the current communities in ND as the final community.

B. Results

Considering that communities detected by AppAcc method for some nodes are empty, when comparing M, AppAcc, Geomod, and AppSLDR algorithms, the average values of metrics are calculated for nodes whose communities detected by AppAcc are not empty. Moreover, the average values of metrics calculated for all selected nodes are also given for M, Geomod, and AppSLDR algorithms. In addition, Geomod was executed on the *Foursquare* dataset for more than a week and still did not terminate, so its results are not given.

1) *Structural cohesiveness*: Metric *communitude* (Section V-A2) is adopted to evaluate the structural cohesiveness of the community. Table III shows the average *communitude* of the communities detected for nodes whose communities detected by AppAcc are not empty. Table IV shows the average *communitude* of the communities detected for all selected nodes.

Table III shows that, in most cases, the AppSLDR algorithm outperforms the AppAcc, M and Geomod algorithms. The *communitude* value of the AppSLDR algorithm is more than 2 times that of the AppAcc method. The AppSLDR algorithm uses the idea of maximizing both goals of the community, which can climb out of a local optimum to find a community with better structural cohesiveness. The M method performs better than AppAcc and Geomod, as it is designed only for

TABLE III: Comparison of *communitude* for nodes whose communities detected by AppAcc are not empty. "Num" means the number of nodes whose communities detected by AppAcc are not empty

Dataset \ Method	M	AppAcc	Geomod	AppSLDR	Num
<i>Brightkite</i>	0.419	0.304	0.449	0.521	77
<i>Gowalla</i>	0.439	0.294	0.431	0.525	76
<i>Flickr</i>	0.262	0.128	0.190	0.279	143
<i>Foursquare</i>	0.406	0.256	/	0.428	53
<i>Syn1</i>	0.352	0.106	0.211	0.349	167
<i>Syn2</i>	0.320	0.080	0.153	0.324	181

* "/" means that the result is not given.

TABLE IV: Comparison of *communitude* for all selected nodes

Dataset \ Method	M	Geomod	AppSLDR
<i>Brightkite</i>	0.510	0.507	0.550
<i>Gowalla</i>	0.529	0.490	0.530
<i>Flickr</i>	0.287	0.198	0.276
<i>Foursquare</i>	0.464	/	0.411
<i>Syn1</i>	0.361	0.211	0.353
<i>Syn2</i>	0.322	0.154	0.325

* "/" means that the result is not given.

the link-based analysis of nodes and does not consider the spatial cohesiveness of the community, so it focuses more on detecting communities with better structural cohesiveness. From the table III, we see that the values of AppAcc is relatively low. The reason is that the AppAcc algorithm guarantees only the closeness of connections within the community without considering the sparsity of the connections inside and outside the community. The performance of Geomod is worse than that of our algorithm. This is because Geomod uses a global modularity (i.e., Q^{geo} [15]) to partition the network into several communities to find the global optimum.

Table IV shows the AppSLDR algorithm is competitive with M method, and better than the Geomod method, which indicates that AppSLDR algorithm does not lose the structural cohesiveness of the community due to the the consideration of spatial cohesiveness. Compared with Table III, we notice some numerical fluctuations in the experimental results on the *Brightkite*, *Gowalla* and *Foursquare* dataset. It is because many nodes with empty community detected by the AppAcc algorithm are not considered in Table III. For example, for the *Brightkite* dataset, Table III shows the average *communitude* of 77 nodes whose communities detected by AppAcc are not empty, and Table IV shows the the average *communitude* of all 200 selected nodes. On the *Syn2* and *Flickr* datasets, the difference between the *communitude* values in Table III and that in Table IV is small.

2) *Spatial cohesiveness*: Here, d_{avg} and d_{IO} are adopted to measure the spatial cohesiveness of the community. Table V shows the average metrics values of the communities detected for nodes whose communities detected by AppAcc are not empty. Table VI shows the the average metrics values of the communities detected for all selected nodes.

Table V shows that on most datasets, the AppSLDR algorithm has smaller d_{avg} and d_{IO} values than other methods,

which means that nodes in the community found by AppSLDR have closer distances. In terms of d_{IO} , AppAcc obtains the best results on the *Foursquare* and *Syn2* datasets. For d_{avg} and d_{IO} , the M method performs worse than the other comparison methods because it does not consider the spatial location information of nodes when detecting community structure. Among the methods that consider the spatial location of nodes, Geomod finds communities with the largest values of d_{avg} and d_{IO} because Geomod detects all communities in the network from the perspective of global optimization. Table VI shows average values of metrics calculated for all selected nodes. Although there are some differences between the values in Table VI and those in Table V, similar conclusions are drawn from Table VI, i.e., the AppSLDR algorithm performs better than the M and Geomod methods.

As seen from Tables III, IV, V and VI, in terms of both structural and spatial cohesiveness, the AppSLDR algorithm outperforms the other comparison methods.

C. Discussion

As mentioned in Section I, the performance of community search algorithms is affected by the parameter k . We use the performance of AppAcc on the *Brightkite* and *Gowalla* datasets to illustrate the effect of k on the final results. A total of 200 given nodes are selected. Fig. 5 shows the number of communities found by the AppAcc algorithm.

1) *Effect of k on the AppAcc algorithm*: For the *Brightkite* dataset, when $k = 1, 5$ and 10 , the number of communities is 200, 67 and 24, respectively. Correspondingly, the number of nodes for which no community is found are 0, 133 and 176. Similar results are obtained on the *Gowalla* dataset. From this phenomenon, we can see that k greatly affects the performance of the AppAcc algorithm. Our algorithm has no parameters.

TABLE V: Comparison of d_{avg} and d_{IO} for nodes whose communities detected by AppAcc are not empty

Dataset \ Method	M		AppAcc		Geomod		AppSLDR	
	d_{avg}	d_{IO}	d_{avg}	d_{IO}	d_{avg}	d_{IO}	d_{avg}	d_{IO}
<i>Brightkite</i>	0.098	0.796	0.024	0.128	0.061	0.378	0.009	0.070
<i>Gowalla</i>	0.045	0.620	0.017	0.108	0.037	0.295	0.005	0.074
<i>Flickr</i>	0.172	0.780	0.036	0.139	0.105	0.415	0.014	0.091
<i>Foursquare</i>	0.066	0.836	0.012	0.075	/	/	0.005	0.094
<i>Syn1</i>	0.422	0.974	0.288	0.705	0.353	0.787	0.285	0.704
<i>Syn2</i>	0.433	1.001	0.240	0.612	0.381	0.861	0.275	0.704

"/" means that the result is not given.

TABLE VI: Comparison of d_{avg} and d_{IO} for all selected nodes

Dataset \ Method	M		Geomod		AppSLDR	
	d_{avg}	d_{IO}	d_{avg}	d_{IO}	d_{avg}	d_{IO}
<i>Brightkite</i>	0.083	0.799	0.062	0.386	0.012	0.107
<i>Gowalla</i>	0.045	0.627	0.037	0.311	0.005	0.105
<i>Flickr</i>	0.164	0.743	0.103	0.408	0.014	0.094
<i>Foursquare</i>	0.076	0.845	/	/	0.021	0.183
<i>Syn1</i>	0.419	0.967	0.355	0.790	0.285	0.702
<i>Syn2</i>	0.435	1.009	0.382	0.863	0.275	0.706

"/" means that the result is not given.

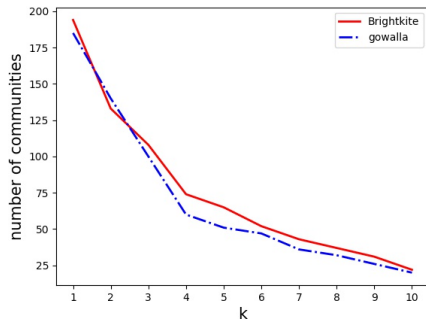


Fig. 5: Number of communities detected by AppAcc as k varies

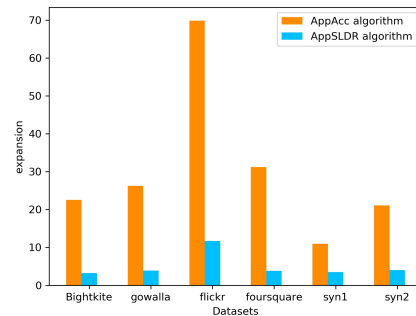


Fig. 6: *expansion* of AppAcc and AppSLDR

For the 200 nodes selected, AppSLDR could find communities for each given node. From this point of view, compared with AppAcc algorithm, AppSLDR is more robust.

Here, *expansion* [48] is adopted to measure the sparsity of external edges of individual communities, calculated as follows:

$$expansion = \frac{|E_C^{out}|}{|C|}, \quad (5)$$

where $|C|$ is the size of community C . The smaller the value of *expansion* is, the better the structural cohesiveness of the community.

2) *Difference between AppAcc and AppSLDR in structural cohesiveness*: Fig. 6 shows that the *expansion* of AppAcc is tens of times larger than that of AppSLDR, which means that the community detected by AppSLDR is more sparsely connected to the external nodes. The reason is that AppAcc considers only the closeness within the community, while AppSLDR considers the closeness within the community as well as the differences inside and outside the community.

3) *Comparison of SLDR and AppSLDR*: We also compare AppSLDR with the SLDR algorithm in terms of runtime, *communitude* and d_{avg} . Due to the slow speed of SLDR, we only compare SLDR and AppSLDR on two datasets, *Syn1* and *Brightkite*. Table VII shows the results of the SLDR and AppSLDR algorithms. We observe that d_{avg} of the SLDR algorithm is slightly better than that of the AppSLDR algorithm. Although AppSLDR loses little spatial cohesiveness, it achieves almost several times faster speed. Moreover, the *communitude* of the AppSLDR algorithm is better than that of the SLDR algorithm.

Specifically, we analyzed the runtime of the algorithms on the 200 nodes. On the *Brightkite* (*Syn1*) dataset, for AppSLDR algorithm, the runtime of 59.5% (90%) nodes is within 10 seconds, and the runtime of 14.5% (3%) nodes is greater than two hours, which had a significant impact on the average results. As a comparison, for SLDR algorithm, only 36% (21.5%) nodes whose runtime is within 10 seconds on the *Brightkite* (*Syn1*) dataset. The reason for the runtime of some nodes more than two hours is as follows: The

TABLE VII: Comparison of SLDR and AppSLDR

Dataset \ Method	SLDR			AppSLDR		
	time(s)	<i>communitude</i>	d_{avg}	time(s)	<i>communitude</i>	d_{avg}
<i>Syn1</i>	2121.0	0.278	0.205	224.6	0.353	0.285
<i>Brightkite</i>	3081.5	0.488	0.007	1408.4	0.550	0.012

degrees of these nodes or their neighbor nodes are greater than several thousand, which causes our algorithms to spend much overhead to generate derived communities.

VI. CONCLUSION

In this paper, we study the SLCD problem, which aims to detect a spatial-aware local community with only local information. To address this problem, we propose the SLDR algorithm and its efficient approximation algorithm called AppSLDR. Extensive experiments on synthetic and real-world datasets demonstrate that the AppSLDR algorithm substantially outperforms other methods in both structural and spatial cohesiveness. In the future, we plan to extend the SLDR and AppSLDR algorithms to detect the community structure in attribute networks.

REFERENCES

- [1] K. Wang, X. Cao, X. Lin, W. Zhang, and L. Qin, "Efficient computing of radius-bounded k -cores," in *Proceedings of the 2018 IEEE 34th international conference on data engineering*. IEEE Computer Society, 2018, pp. 233–244.
- [2] Q. Zhu, H. Hu, C. Xu, J. Xu, and W.-C. Lee, "Geo-social group queries with minimum acquaintance constraints," *The VLDB Journal*, vol. 26, no. 5, pp. 709–727, 2017.
- [3] Y. Fang, R. Cheng, X. Li, S. Luo, and J. Hu, "Effective community search over large spatial graphs," *Proceedings of the VLDB Endowment*, vol. 10, no. 6, pp. 709–720, 2017.
- [4] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical review E*, vol. 69, no. 6, p. 066133, 2004.
- [5] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, 2009.
- [6] L. Chen, C. Liu, R. Zhou, J. Li, X. Yang, and B. Wang, "Maximum co-located community search in large scale social networks," *Proceedings of the VLDB Endowment*, vol. 11, no. 10, pp. 1233–1246, 2018.
- [7] W. Luo, D. Zhang, L. Ni, and N. Lu, "Multiscale local community detection in social networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 3, pp. 1102–1112, 2021.
- [8] M. Sozio and A. Gionis, "The community-search problem and how to plan a successful cocktail party," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 939–948.
- [9] K. Yao and L. Chang, "Efficient size-bounded community search over large networks," *Proceedings of the VLDB Endowment*, vol. 14, no. 8, pp. 1441–1453, 2021.
- [10] P. Li, L. Huang, C. Wang, J. Lai, and D. Huang, "Community detection by motif-aware label propagation," *ACM Transactions on Knowledge Discovery from Data*, vol. 14, no. 2, pp. 22:1–22:19, 2020.
- [11] K. Nath, R. Shanmugam, and V. Vijayakumar, "ma-code: A multi-phase approach on community detection in evolving networks," *Information Sciences*, vol. 569, pp. 326–343, 2021.
- [12] R. Guimera and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *nature*, vol. 433, no. 7028, pp. 895–900, 2005.
- [13] D. Guo, "Regionalization with dynamically constrained agglomerative clustering and partitioning (redcap)," *International Journal of Geographical Information Science*, vol. 22, no. 7, pp. 801–823, 2008.
- [14] P. Expert, T. S. Evans, V. D. Blondel, and R. Lambiotte, "Uncovering space-independent communities in spatial networks," *Proceedings of the National Academy of Sciences*, vol. 108, no. 19, pp. 7663–7668, 2011.
- [15] Y. Chen, J. Xu, and M. Xu, "Finding community structure in spatially constrained complex networks," *International Journal of Geographical Information Science*, vol. 29, no. 6, pp. 889–911, 2015.
- [16] L. Wang, Y. Zeng, Y. Li, Z. Liu, J. Ma, and X. Zhu, "Research on resolution limit of community detection in location-based social networks," in *Proceedings of the 2019 International Conference on Networking and Network Applications*. IEEE, 2019, pp. 90–95.
- [17] F. Zhang, Y. Zhang, L. Qin, W. Zhang, and X. Lin, "When engagement meets similarity: Efficient (k, r) -core computation on social networks," *Proceedings of the VLDB Endowment*, vol. 10, no. 10, pp. 998–1009, 2017.
- [18] W. Luo, N. Lu, L. Ni, W. Zhu, and W. Ding, "Local community detection by the nearest nodes with greater centrality," *Information Sciences*, vol. 517, pp. 377–392, 2020.
- [19] A. Clauset, "Finding local community structure in networks," *Physical review E*, vol. 72, no. 2, p. 026132, 2005.
- [20] F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," *Web Intelligence and Agent Systems: An International Journal*, vol. 6, no. 4, pp. 387–400, 2008.
- [21] D. Luo, Y. Bian, Y. Yan, X. Liu, J. Huan, and X. Zhang, "Local community detection in multiple networks," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. ACM, 2020, pp. 266–274.
- [22] W. Cui, Y. Xiao, H. Wang, and W. Wang, "Local search of communities in large graphs," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 991–1002.
- [23] B. Viswanath, A. Post, K. P. Gummadi, and A. Mislove, "An analysis of social network-based sybil defenses," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 363–374, 2010.
- [24] M. Barthélemy, "Spatial networks," *Physics Reports*, vol. 499, no. 1-3, pp. 1–101, 2011.
- [25] J. Luo, X. Cao, X. Xie, Q. Qu, Z. Xu, and C. S. Jensen, "Efficient attribute-constrained co-located community search," in *Proceedings of the 36th IEEE International Conference on Data Engineering*. Dallas, TX, USA: IEEE, 2020, pp. 1201–1212.
- [26] Y. Fang, Z. Wang, R. Cheng, X. Li, S. Luo, J. Hu, and X. Chen, "On spatial-aware community search," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 783–798, 2019.
- [27] Q. Liu, Y. Zhu, M. Zhao, X. Huang, J. Xu, and Y. Gao, "VAC: vertex-centric attributed community search," in *Proceedings of the 36th IEEE International Conference on Data Engineering*. Dallas, TX, USA: IEEE, 2020, pp. 937–948.
- [28] K. He, P. Shi, D. Bindel, and J. E. Hopcroft, "Krylov subspace approximation for local community detection in large networks," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 5, pp. 1–30, 2019.
- [29] J. Zhu, B. Chen, and Y. Zeng, "Community detection based on modularity and k -plexes," *Information Sciences*, vol. 513, pp. 127–142, 2020.
- [30] C. Pizzuti, "Evolutionary computation for community detection in networks: A review," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 464–483, 2018.
- [31] M. Okuda, S. Satoh, Y. Sato, and Y. Kidawara, "Community detection using restrained random-walk similarity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 89–103, 2021.
- [32] X. Luo, Z. Liu, M. Shang, J. Lou, and M. Zhou, "Highly-accurate community detection via pointwise mutual information-incorporated symmetric non-negative matrix factorization," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 463–476, 2021.
- [33] C. Pizzuti, "A multiobjective genetic algorithm to find communities in complex networks," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 3, pp. 418–430, 2012.
- [34] M. Li, S. Lu, L. Zhang, Y. Zhang, and B. Zhang, "A community detection method for social network based on community embedding," *IEEE Transactions on Computational Social Systems*, vol. 8, no. 2, pp. 308–318, 2021.
- [35] Y. Li, K. He, K. Kloster, D. Bindel, and J. E. Hopcroft, "Local spectral clustering for overlapping community detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 12, no. 2, pp. 17:1–17:27, 2018.

- [36] L. Ni, W. Luo, W. Zhu, and B. Hua, "Local overlapping community detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 14, no. 1, pp. 3:1–3:25, 2020.
- [37] C. Lyu, Y. Shi, and L. Sun, "A novel local community detection method using evolutionary computation," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3348–3360, 2021.
- [38] F. Cheng, C. Wang, X. Zhang, and Y. Yang, "A local-neighborhood information based overlapping community detection algorithm for large-scale complex networks," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 543–556, 2021.
- [39] K. Wang, W. Zhang, X. Lin, Y. Zhang, L. Qin, and Y. Zhang, "Efficient and effective community search on large-scale bipartite graphs," in *Proceedings of the 37th IEEE International Conference on Data Engineering*. Chania, Greece: IEEE, 2021, pp. 85–96.
- [40] X. Huang, H. Cheng, L. Qin, W. Tian, and J. X. Yu, "Querying k-truss community in large and dynamic graphs," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 1311–1322.
- [41] B. Liu, F. Zhang, W. Zhang, X. Lin, and Y. Zhang, "Efficient community search with size constraint," in *Proceedings of the 37th IEEE International Conference on Data Engineering*. Chania, Greece: IEEE, 2021, pp. 97–108.
- [42] J. Kim, T. Guo, K. Feng, G. Cong, A. Khan, and F. M. Choudhury, "Densely connected user community and location cluster search in location-based social networks," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. ACM, 2020, pp. 2199–2209.
- [43] W. Cui, Y. Xiao, H. Wang, Y. Lu, and W. Wang, "Online search of overlapping communities," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 2013, pp. 277–288.
- [44] M. Gong, L. Ma, Q. Zhang, and L. Jiao, "Community detection in networks by using multiobjective evolutionary algorithm with decomposition," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 15, pp. 4050–4060, 2012.
- [45] F. Folino and C. Pizzuti, "An evolutionary multiobjective approach for community discovery in dynamic networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1838–1852, 2014.
- [46] M. Liu, W. Zeng, and J. Zhao, "A fast bi-objective non-dominated sorting algorithm," *Pattern Recognition and Artificial Intelligence*, vol. 24, no. 4, pp. 538–547, 2011.
- [47] A. Miyauchi and Y. Kawase, "What is a network community? a novel quality function and detection algorithms," in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 1471–1480.
- [48] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly, "Metrics for community analysis: A survey," *ACM Computing Surveys*, vol. 50, no. 4, pp. 1–37, 2017.