# K-Hop Community Search Based On Local Distance Dynamics

**Tao Meng[1], Lijun Cai[1], Tingqin He[1], Lei Chen[2] and Ziyun Deng[3]**
[1] College of Information Science and Engineering, Hunan University
Chang Sha, 410082 - China
[e-mail: mengtao, ljcai, hetingqin@hnu.edu.cn]
[2] College of Electrical and Information Engineering, Hunan University
Chang Sha, 410082 – China
[e-mail: chenleixyz123@hnu.edu.cn]
[3] Department of Economics and Trade, ChangSha Commerce and Tourism College
Chang Sha, 410082 - China
[e-mail: dengziyun@126.com]
*Corresponding author: Lijun Cai

*Received November 12, 2017; revised February 7, 2018; accepted March 15, 2018; published July 31, 2018*

## *Abstract*

Community search aims at finding a meaningful community that contains the query node and also maximizes (minimizes) a goodness metric. This problem has recently drawn intense research interest. However, most metric-based algorithms tend to include irrelevant subgraphs in the identified community. Apart from the user-defined metric algorithm, how can we search the natural community that the query node belongs to? In this paper, we propose a novel community search algorithm based on the concept of the k-hop and local distance dynamics model, which can naturally capture a community that contains the query node. The basic idea is to envision the nodes that k-hop away from the query node as an adaptive local dynamical system, where each node only interacts with its local topological structure. Relying on a proposed local distance dynamics model, the distances among nodes change over time, where the nodes sharing the same community with the query node tend to gradually move together, while other nodes stay far away from each other. Such interplay eventually leads to a steady distribution of distances, and a meaningful community is naturally found. Extensive experiments show that our community search algorithm has good performance relative to several state-of-the-art algorithms.

***Keywords:*** Community Search, complex network, k-hop

## 1. Introduction

Most complex networks in nature and human society, such as social networks and communication networks, contain community structures. The goal of community detection is to identify all communities in the entire network, which is a fundamental graph-mining task that has been widely studied in the literature [1, 2, 3, 4, 5]. Recently, a different but related problem called community search has been studied, which is to find the most likely community that contains the query node [6]. It has a wide range of applications in complex networks analysis, such as social contagion modeling [7] and social circle detection [8]. As one example, in a social network, given a group of people who have been exposed to a highly infectious disease, who will most likely be affected? Obviously, we may want to monitor people in their local community [9]. In this paper, we study the modeling and searching of the community of a query node.

In previous studies on these problems, a goodness metric is usually used to identify whether a subgraph forms a community. Many approaches have been proposed to find a subgraph that contains the query node, and the goodness metric is maximized or minimized, such as k-core [6, 9, 10], k-truss [11, 12] and densest graph [13]. However, most existing goodness metrics do not address the "free rider effect" issue; that is, nodes irrelevant to the query node or far away from it are included in the identified community [11]. As an example, suppose that we use the widespread k-core as a goodness metric [9], which requires each node to be connected with at least k nodes in the target subgraph. Consider **Fig. 1**; for query node 1, the subgraph A shaded grey is a 3-core containing the query node. However, it includes subgraph C, which is not relevant to the query node. Intuitively, subgraph B should be the best search result. We refer to irrelevant subgraphs, such as C, as a free rider in a community search. Moreover, real-world applications often generate massive-scale graphs and require efficient processing. Therefore, achieving strong scalability together with a high-quality community search remains a challenging, open research problem.

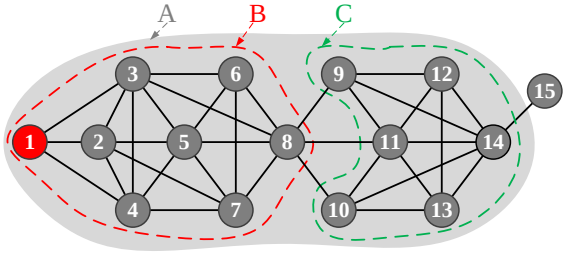



**Fig. 1.** An example of a community search with a free rider.

In this paper, instead of introducing a new goodness metric for community searches like k-core [9] or k-truss [10], we consider the problem of a community search from a new point view: local distance dynamics. We will demonstrate that a community search in this intuitive way has many attractive benefits, but let us first illustrate the basic idea.

### 1.1 Motivation

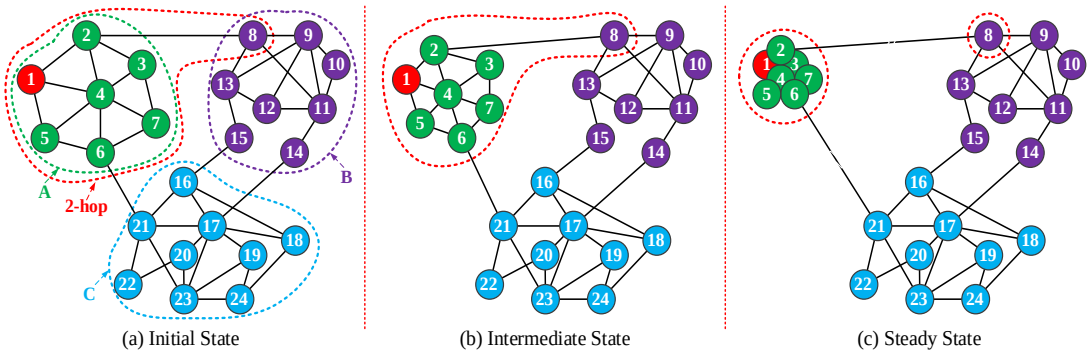
A recent study by Shao et al. proposed a novel algorithm for community detection - the Attractor algorithm [3]. This algorithm views the entire graph as an adaptive global dynamical system and updates its dynamics over time. The process of the global distance dynamic model involves the following stages: First, each edge is associated with an initial distance. Then, in a

sequential process, each distance gradually shrinks or stretches by the interaction with its local topological structure. Finally, all distance convergence results in 0 or 1. As a result, all communities and outliers are naturally popped by removing the edges with distances equivalent to 1.

Inspired by the Attractor algorithm, a straightforward strategy for community search could be a global distance dynamics approach like Attractor. However, global distance dynamics is costly as it needs to explore the entire graph before finding the target community for the query node. This is unacceptable for large graphs. In fact, it should not be necessary to involve the entire graph in the community search. Moreover, the query node is not considered since its focus is not a community search but community detection.

In this paper, we propose a novel community search algorithm based on the concept of the k-hop and local distance dynamics model called the K-Hop model, which can naturally capture a community that contains the query node. The basic idea is to envision the nodes that k-hop away from a query node as an adaptive local dynamical system, where each node only interacts with its local topological structure. Relying on a proposed local distance dynamics model, the distances among nodes change over time, where the nodes sharing the same community with the query node tend to gradually move together, while other nodes stay far away from each other. Such interplay eventually leads to a steady distribution of distances, and a meaningful community is naturally found.

To better illustrate the basic idea, let us take a simple network as an example. **Fig. 2** displays three snapshots of the simulated local distance dynamics model on an artificial network. In this network, there exist three subgraphs with different colors. Suppose that the query node is the red node in subgraph A and k=2. **Fig. 2(a)** shows the initial state of the network. First, starting from the query node, the nodes that are a 2-hop away from it can form a 2-hop subgraph. For any two connected nodes, start with an initial distance value in the 2-hop subgraph. **Fig. 2(b)** displays the intermediate state of the network. Due to the influence of its neighbors, the distances among nodes sharing the same community with the query node tend to decrease, while those in irrelevant communities increase. **Fig. 2(c)** depicts the steady state of the network. Finally, as time evolves, all distances converge to 0 or 1 in the 2-hop subgraph. Therefore, the target community is naturally popped by removing the edges with distances equivalent to 1.




(a) Initial State (b) Intermediate State (c) Steady State

**Fig. 2.** The illustration of a k-hop community search based on local distance dynamics.

## 1.2 Contribution

The major benefits of this paper are as follows:

- **Natural Community Search:** Instead of maximizing or minimizing user-defined metrics, K-Hop considers the problem of the community search from a new point view: local

distance dynamics. Relying on a proposed local distance dynamics model, K-Hop can naturally capture a community that contains the query node.

- **Cohesive Community:** We analyze the structural and computational properties of K-Hop and show that it effectively avoids the free rider effect, faithfully searching the cohesive community.
- **Line Time Algorithm:** Thanks to the local distance dynamics model, K-Hop only needs to update the distances of linked nodes in the k-hop subgraph over time, which make the local distance dynamics model computationally tractable and efficient. This property of K-Hop lends itself to large-scale graphs.
- **Extensive Experiments:** Extensive experiments on both synthetic networks and a variety of large real-world networks demonstrate the effectiveness and efficiency of our community model and search algorithm.

The remainder of the paper is organized as follows: Related work is described in section 2. Section 3 describes the background of this paper. Section 4 shows our algorithm in detail. Extensive experimental evaluation is presented in section 5. Section 6 provides a brief conclusion.

## 2. Related Work

The work presented in this paper is closely related to community detection, cohesive subgraph mining and community search.

**Community Detection.** The goal of community detection is to identify all communities in an entire network [14]. Generally, community detection can fall into two major categories: non-overlapping community detection [2, 3, 9] and overlapping community detection [15, 16]. These problems are extensively studied in the literature. For detailed reviews of community detection, please refer to [17]. All these methods consider communities with global information about the entire network structure. However, it is very difficult for us to get the entire network structure nowadays, in particular for large networks, such as social graph and web graph [18]. In order to circumvent this problem, community search was proposed and has attracted much attention.

**Cohesive Subgraph Mining.** The cohesive subgraph mining problem aims at enumerating all cohesive subgraphs from an entire graph. There are many different definitions of cohesive subgraphs in the literature, including maximal clique [19], k-core [20], k-truss [21], and maximal k-edge connected subgraph [22]. However, none of these works consider the query node, which was first proposed in [6]. Thus, cohesive subgraph mining is significantly different from the query driven community search.

**Community Search.** The community search focuses on finding a community from a query node only with the local network structure. In recent years, several metric-based community search models have been studied, such as k-core [9], k-truss [12], and k-influential community [10]. The k-core community is based on the k-core subgraph, which contains nodes with at least k edges. The k-truss community is based on the k-truss subgraph, where every edge is contained in at least (k-2) triangles and connected by triangles. The k-influential community is based on the concept of k-core and maximizes the influential of the target community. However, most existing algorithms tend to include irrelevant subgraphs in the identified community, or they experience computational bottleneck.

In summary, previous approaches mainly focus on optimizing user-defined metrics; here, we propose an intuitive way to search the community structure based on local distance dynamics, which not only captures high-quality communities but also makes it possible to

handle large-scale networks. Moreover, to the best of our knowledge, this is the first work that proposes the local distance dynamics model in community search.

## 3. Preliminaries

In this paper, we focus on an undirected and unweighted simple graph *G=(V, E)*, where *V* and *E* are a set of nodes and edges, respectively. Other types of graphs, such as directed and weighted, can be handled with only slight modifications.

The structure of a node can be described be its neighbors and the distance between two nodes according to how they share neighbors. The neighbors of a node are a node set composed of all its adjacent nodes and the node itself.

**Definition 1 (neighbors of node u).** Given an undirected graph *G=(V, E)*, the neighbors of node *u* are denoted by *N*(*u*) and defined as follows:

$$N(u)=\{v\in V|\{u,v\}\in E\}\cup\{u\} \tag{1}$$

The distance of adjacent nodes is computed by the common nodes in the structural neighborhoods. This measurement is called the Jaccard distance and is defined as follows:

**Definition 2 (Jaccard Distance).**Given an undirected graph *G=(V, E)*, the Jaccard distance between node *u* and node *v* is defined as:

$$d(u,v)=1-\frac{|N(u)\cap N(v)|}{|N(u)\cup N(v)|} \tag{2}$$

In the above equation, | * |indicates the node number of set *, and *N(u)* is the neighbors of node *u*.

For the weighted undirected graph, because each edge has a different weight, the compute model of the Jaccard distance is different, and the new compute model is further extended as:

$$d(u,v)=1-\frac{\sum_{x\in N(u)\cap N(v)}\left(w(u,x)+w(v,x)\right)}{\sum_{\{x,y\}\in E;x,y\in N(u)\cup N(v)}w(x,y)} \tag{3}$$

## 4. K-Hop Community Search Based On Local Distance Dynamics

### 4.1 K-Hop Community Search

Our goal is to find a target community for the query node from a large-scale graph within a short computation time. However, the global distance dynamics is costly, as it needs to visit the entire graph. This is unacceptable for large-scale graphs. Therefore, the reasonable scope of the target community needs to be considered first.

In order to efficiently identify the scope of the community that the query node belongs to, we use an observation of real-world graphs: The best community for a given node is in the neighborhood of the node. This observation is based on a well-known property of real-world graphs: the small world effect, which is the name given to the finding that the average path (hop) between vertices in a network is small, and the local structure of a network still has obvious grouping characteristics [23, 24, 25]. That is to say, nodes that have a long path (hop) length to the query node are unlikely to share the same clique with the query node. For example, in a social network, if two users do not have direct friendships with each other and have no common friends between them, they are unlikely to be in the same community.

Based on this property, K-Hop can prune the distance evaluation for the nodes that are more than a k-hop away from the query node. Specifically, K-Hop first roughly detects a k-hop subgraph by searching the nodes that are a k-hop away from the query node. It then refines the k-hop subgraph to find the best community that the query node belongs to by the local distance dynamics model. Before proceeding further, we give the formal definition of the k-hop subgraph as follows.

**Definition 3 (K-Hop Subgraph).** Given a graph $G=(V, E)$, a query node $q \in V$ and an integer k>0, $G'$ is a k-hop subgraph if and only if $G'$ is connected, and each node $u$ in $G'$ is at most a k-hop away from node $q$.

**Fig. 3** shows an example of a k-hop subgraph of the query node $q$. In this example, $q$ is the query node, and A, B and C are a 1-hop subgraph, 2-hop subgraph and 3-hop subgraph, respectively. The 1-hop subgraph has nodes $\{q, h_{11}, h_{12}, h_{13}, h_{14}, h_{15}, h_{16}\}$, and its nodes are 1 hop away from the query node $q$.

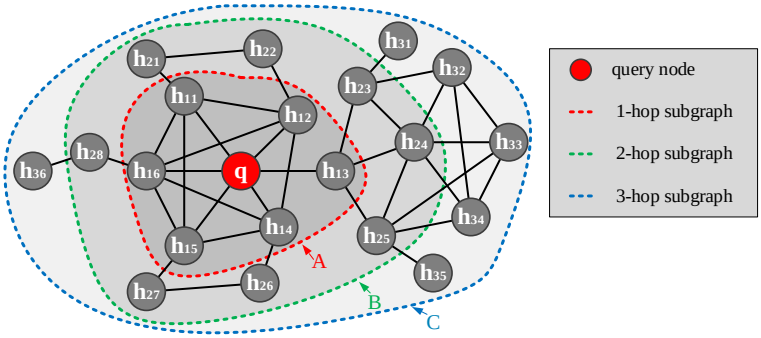



**Fig. 3.** An example graph for k-hop community.

On the basis of the definitions of the k-hop subgraph, we give the definition of the k-hop community as follows, where the parameter k controls the scope of the community.

**Definition 4 (K-Hop Community).** Given a graph $G=(V, E)$, a query node $q \in V$ and an integer k>0, $C$ is a k-hop community if $C$ satisfies the following constrains.

- **Connectivity.** $C$ is a connected k-hop subgraph and contained node $q$.
- **Cohesiveness.** Employing the proposed local distance dynamics model on the k-hop subgraph, where nodes in the target community move together and other nodes keep far away from the query node $q$.

Clearly, the *connectivity* constraint requires that the k-hop community containing the query node $q$ be connected. In addition, the *cohesiveness* constraint ensures that each node is as close as possible to the query node in the k-hop community. With the connectivity and cohesiveness constraints, we can ensure that the k-hop community is a connected and cohesive subgraph. The following example illustrates the definition of a k-hop community.

Let us reconsider the graph shown in **Fig. 3**. Assume that k = 1 and $q$ is the query node. By definition 3, we can see that the 1-hop subgraph is included by node set $\{q, h_{11}, h_{12}, h_{13}, h_{14}, h_{15}, h_{16}\}$. However, it includes node $h_{13}$, which is intuitively not relevant to the query node. Relying on a proposed local distance dynamics model, the nodes $q$, $h_{11}$, $h_{12}$, $h_{14}$, $h_{15}$ and $h_{16}$ are moving together, while node $h_{13}$ stays far away from them. As a result, the target community for node $q$ is $C = \{q, h_{11}, h_{12}, h_{14}, h_{15}, h_{16}\}$.

**Problem Definition.** The problem of the k-hop community search studied in this paper is defined as follows. Given a graph *G=(V, E)*, a query node $q \in V$ and an integer k>0, find a k-hop community containing *q*.

### 4.2 Local Distance Dynamics Model

After specifying the scope of the target community, the next crucial step is to determine the interaction model among nodes in the k-hop subgraph to simulate the distance dynamics. The whole local distance dynamics model contains two parts: the core edge interaction model and border edge interaction model. If one edge is a core edge, through the core edge interaction model, this edge will have a final distance value (0 or 1). In contrast with the core edge, the border edge will not have a final distance value; the goal of the border edge interaction model is to make the core edge interact more fully. By removing the core edges with distances equivalent to 1, the target community is naturally popped. In the following, we will elaborate how the distance changes in the two different interaction models.

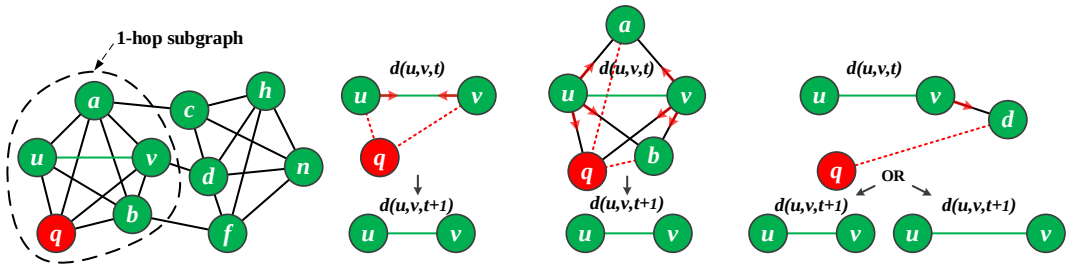
(1) *Core Edge Interaction Model*

If an edge is a core edge, then two endpoints of this edge must be contained in the k-hop subgraph. We define the core edge as follows.

**Definition 5 (Core Edge).** Given a k-hop subgraph $G'(V', E') \subseteq G(V, E)$, the edge *e*={*u*, *v*} $e = \{u, v\} \in E$ is a core edge *iff* *e* ={*u*, *v*} $\in E'$.

As shown in **Fig. 4(a)**, in this example network, node $q \in V$ is the query node and k=1, and the dashed circle denotes the 1-hop subgraph. According to definition 5, the edges *e(u,v)*, *e(u,a)*, *e(u,b)*, *e(u,q)*, *e(v,a)*, *e(v,b)*, *e(v,q)*, *e(a,b)*, *e(a,q)* and *e(b,q)* are core edges since they are contained in the 1-hop subgraph.

Formally, let *e*={*u*, *v*} $\in E$ be a core edge between two adjacent nodes *u* and *v*, and *d*(*u*,*v*) is its initial distance. Obviously, any change in distance *d(u,v)* actually results from the variation of node *u* and node *v*. Relying on its complete local topological structure (see **Fig. 4**), there are three distinct interaction patterns that allow influencing the distance *d(u,v)*.




(a) An Example Network (k=1, *q* is the query node) (b) Influence from direct linked nodes (c) Influence from common neighbors (d) Influence from exclusive neighbors

**Fig. 4.** Local distance dynamics of one core edge.

**Local Pattern 1.** Here, we consider the first interaction pattern: influence from direct linked nodes *u* and *v* (see **Fig. 4(b)**). Through mutual interactions, one node attracts another to move toward itself and thus leads to a decrease in distance *d(u,v)*. Formally, we define the change in *d(u,v)* from the influence of the direct linked nodes, *DI*, as follows:

$$DI = -\left( \frac{f\left(1-d\left(u,v\right)\right).\left(1-d\left(u,q\right)\right)}{deg\left(u\right)} + \frac{f\left(1-d\left(u,v\right)\right).\left(1-d\left(v,q\right)\right)}{deg\left(v\right)} \right) \tag{4}$$

In pattern *DI*, *f(·)* is a coupling function and *f(.)=sin(.)* is used in this study. The term *1-d(u,v)* implies the similarity between two direct linked nodes *u* and *v*, and the greater the similarity between the two nodes, the greater is their influence. The term *1-d(u,q)* indicates the similarity between node *u* and query node *q*; the more similar it is to the query node, the higher the influence node *u* will have. The term *1/deg(u)* is a normalized factor used to consider the different influences between linked nodes with diverse degrees. The nodes with fewer links are easier to be influenced compared to the nodes with more links.

Take a friendship network as an example. In general, each person affects the people they know and tend to increase their cohesiveness gradually. Moreover, the more similar the two people are, the greater the influence they will have between each other; the more similar they are to the query people, the more likely they are to share the same community with the query people; people with more friends are harder to influence compared to people with fewer friends.

**Local Pattern 2.** The second interaction pattern happens when common neighbors exist between nodes *u* and *v* (see **Fig. 4(c)**). The common neighbors between node *u* and *v* are denoted by *CN=(N(u)-u)∩(N(v)-v)*. As the common neighbors have both links with node *u* and *v*, they attract the two nodes to move towards them, and there is a decrease in distance *d(u,v)*. Formally, to characterize the change in distance *d(u,v)*, we define *CI*, indicating the influence from the interactions of common neighbors, as follows:

$$CI = -\sum_{x \in CN} \left( \begin{array}{l} \frac{1}{deg(u)} \cdot f\left(1-d\left(x,u\right)\right) \cdot \left(1-d\left(x,v\right)\right) \\ \qquad + \frac{1}{deg(v)} \cdot f\left(1-d\left(x,v\right)\right) \cdot \left(1-d\left(x,u\right)\right) \end{array} \right) \cdot \left(1-d\left(x,q\right)\right) \tag{5}$$

In pattern *CI*, the two terms *1-d(x,u)* and *1-d(x,v)* indicate the similarity of common neighbor *x* to node *u* and *v*. If *x* is more similar to *u*, the influence of *x* on *v* is more similar to the influence from *u*. The term *1-d(x,q)* implies similarity between common neighbor *x* and query node *q*; the more similar it is to the query node, the greater the influence common node *x* will have.

Let us reconsider the friendship network as an example. Obviously, when two people share many common friends, their degree of similarity becomes large, and this tends to gradually increase their cohesiveness. Furthermore, if their common friends are close to the query people, they tend to share the same community with the query people.

**Local Pattern 3.** The influence from exclusive neighbors is the third interaction pattern (see **Fig. 4(d)**). The exclusive neighbors only belong to node *u* or *v* and are donated by *EN(u)=N(u)-(N(u)∩N(v))* and *EN(v)=N(v)-(N(u)∩N(v)),* respectively. In this pattern, each exclusive neighbor may have a positive or negative influence on distance *d(u,v)*. To determine the positive or negative influence of exclusive neighbors on the distance, a similarity-based heuristic strategy is proposed. The basic idea is to investigate whether each exclusive neighbor of node *u* is similar to query node *q* and vice versa. If the exclusive neighbor of node *u* is similar to query node *q*, the movement of node *u* toward the exclusive neighbor results in the decrease in distance *d(u,v)*. Similarly, if the exclusive neighbor is not similar to query node *q*, the movement of node *u* toward the exclusive neighbor results in the increase in distance *d(u,v)*. Formally, we define the degree of positive or negative influence on distance *d(u,v)* from the exclusive neighbor as follows:

$$\sigma\left(x,q\right) = \begin{cases} \left(1-d\left(x,q\right)\right) & \left(1-d\left(x,q\right)\right) \geq \lambda \\ \left(1-d\left(x,q\right)\right) - \lambda & \text{otherwise} \end{cases} \tag{6}$$

In equation 6 above, the term $\lambda$ is a cohesive parameter and will be further discussed in section 5.2. Then, we define the change of *d(u,v)* from the influence of exclusive neighbors, *EI*, as follows:

$$EI = \begin{pmatrix} -\sum_{x \in EN(u)} \left( \frac{1}{deg(u)} \cdot f\left(1 - d(x,u)\right) \cdot \sigma(x,q) \right) \\ -\sum_{y \in EN(v)} \left( \frac{1}{deg(v)} \cdot f\left(1 - d(y,v)\right) \cdot \sigma(y,q) \right) \end{pmatrix} \tag{7}$$

Finally, by considering three interaction patterns together, the dynamics of distance *d(u,v)* on core edge *e(u, v)* over time is governed by:

$$d(u,v,t+1) = d(u,v,t) + DI(t) + CI(t) + EI(t) \tag{8}$$

In equation 8 above, the term *d(u,v,t+1)* is the new distance at time step *t+1*. *DI(t)*, *CI(t)* and *EI(t)* are three different influences from the directed nodes, common neighbors, and exclusive neighbors on distance *d(u,v,t)* at time step *t*.

(2) *Border Edge Interaction Model*

By using three interaction patterns (*DI*, *CI*, and *EI*), the dynamics distance of each core edge is simulated by the core edge interaction model. In the core edge interaction model, before computing *CI* or *EI*, we need to measure the similarity between the direct linked node and the common neighbors or the exclusive neighbors over time. However, the common neighbors or the exclusive neighbors of the direct linked node may not be contained in the k-hop subgraph. Therefore, how to simulate the distance dynamics of the edge between the nodes in the k-hop subgraph and the nodes not contained in the k-hop subgraph is also important. In this study, we call this edge the border edge. Before proceeding further, we give the formal definition of the border edge as follows.

**Definition 6 (Border Edge).** Given a k-hop subgraph $G'(V', E') \subseteq G(V, E)$, the edge $e=\{v, d\} \in E$ is a border edge *iff* $v \in V'$ and $d \notin V'$, and vice versa.

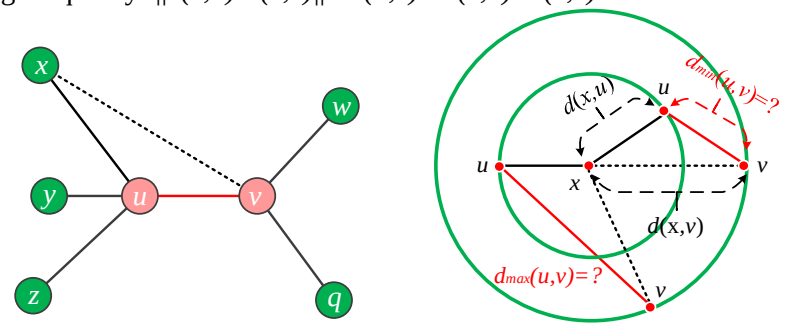
Formally, let $e=\{v, d\} \in E$ be a border edge between two adjacent nodes *v* and *d*, and assume node *v* is contained in the k-hop subgraph and node *d* does not belong to the k-hop subgraph. Unlike the core edges, the change in distance *d(v,d)* cannot result from the variation in node *v* and node *d*; the local topological structure of node *d* is out of the interaction range of the local distance dynamics model. In order to simulate the dynamics of each border edge, here we proposed a border edge interaction model to estimate the potential change in distance *d(v,d)*.

The basic idea of the border edge interaction model is as follows. In the core edge interaction model, the Jaccard distance [26] is selected as a metric to measure the distance of a core edge in the k-hop subgraph. The Jaccard distance has several nice properties; it is non-negative and symmetric, and it is characterized by the triangular inequality. In all the properties of the Jaccard distance, the triangular inequality rule is very important and is proved in different ways [27]. The triangular inequality rule enables estimating the minimum and maximum distances of two sets *B*, *C* without having to directly calculate it if their pairwise distance to a third set *A* is given. The complete triangular inequality rule of Jaccard distance is as follows:

$$\left\| d(A,B) - d(A,C) \right\| \le d(B,C) \le d(A,B) + d(A,C) \tag{9}$$

Where ||*|| is the absolute value function. To describe the triangular inequality more clearly, **Fig. 5** shows an example. In **Fig. 5(a)**, we are interested in estimating or bounding the distance *d(u,v)* by using the existing distance *d(x,u)* and *d(x,v)*. According to the triangular inequality

rule of the Jaccard distance (see **Fig. 5(b)**), in triangle $\Delta xuv$, the distance $d(u,v)$ clearly satisfies the following inequality: $||d(x,u)-d(x,v)|| \le d(u,v) \le d(x,u)+d(x,v)$.

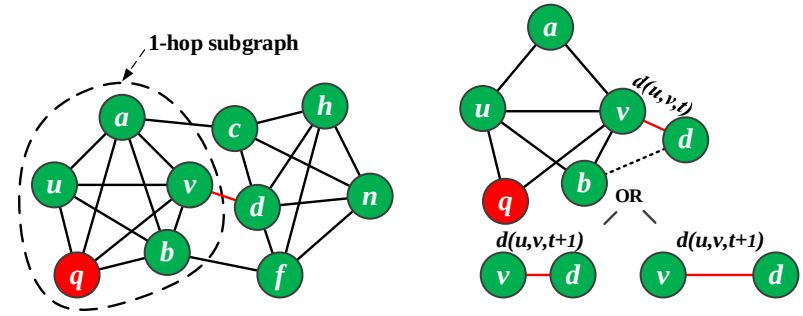



(a) An example network (b) The idea of triangle distance.

**Fig. 5.** Triangular inequality rule of Jaccard distance.

Based on the triangular inequality rule of Jaccard distance, it is possible to quickly predict the distance of the border edge over time. However, the border edge can be contained in multiple triangles (see **Fig. 6**), so the border edge interaction model is defined as follows:

$$d(v,d) = \frac{1}{2} * \left( \left( \left\| d(x,v) - d(x,d) \right\| \right) + \left( d(x,v) + d(x,d) \right) \right) \tag{10}$$




(a) An Example Graph (k=1, *q* is the query node) (b) The border edge interaction model

**Fig. 6.** Local distance dynamics of one border edge.

Where node $x$ is any neighbor of node $v$ and contained in the k-hop subgraph. The distance of the border edge is the average value of the triangular inequality distance between node $x$ and node $u$, $v$. By using the border edge interaction model, we can accurately estimate the dynamics distance of any border edge and do not need extra computation; this is very important for speeding up the local distance dynamics model under large-scale networks.

(3) *Free Rider Effect*

Since our k-hop community search model is based on the concept of the k-hop, the communities capture good structural properties, such as connected and bounded diameters. In addition, since the k-hop subgraph uses the local dynamics distance model to simulate the distance dynamics, it also has a cohesive structure. As a result, K-Hop avoids the "free rider effect" [12, 13].

**Structure Property: Bounded Diameter in K-Hop Community.** The diameter of a community is considered an important feature of a good community [28]. The diameter of a connected k-hop community with *n* nodes is no more than 2k, independent of the size of *n*, and usually produces a good result with value 1 or 2. This property guarantees that each node is as close as possible to every other node in the community, including the query node.

**LEMMA 1.** Suppose that a k-hop community contains *n* nodes and has diameter *d*. Then, the diameter of the k-hop community always satisfies $d \leq 2k$.

**Proof Sketch:** We prove the lemma by contradiction. Assume that the diameter of the k-hop community is more than 2k. Then, following definition 4, every node in the k-hop community is at most a k-hop away from the query node, and every node in the k-hop community is also a structure connected to the query node. Thus, all nodes in the k-hop community have a distance of at most 2k hop. This contradicts the fact that the diameter of the k-hop community is more than 2k.

Consider the 1-hop community in **Fig. 6(a)** as an example. The diameter of the 1-hop community is 1, which is much less than the diameter upper bound 2*1=2.

**Computational Property: Cohesive K-Hop Community.** This property ensures the high connectivity of a community, which was proposed as a criterion of a good community in [13]. Relying on the local intrinsic topological structure, the dynamics of each distance is simulated according to the local distance dynamics model. As time evolves, the nodes in the target community move together, while other nodes remain far away from the query node.

(4) *Algorithem K-Hop*

In this section, we give a comprehensive description of the K-Hop algorithm. The K-Hop process is very simple, consisting mainly of the following three steps.

**Step 1, initialization of interaction scope and distances**. First, K-Hop classifies the edges that belong to the interaction scope of the target community as either a core edge or border edge. At the initial time, each edge in the core edge set or border edge set is only associated with an initial distance, without any interaction.

**Step 2, dynamic interaction**. As time evolves, relying on the local intrinsic topological structure, the distance dynamics of each edge in the interaction scope is simulated according to the proposed core edge interaction model and border edge interaction model. As a result, the nodes in the target community move together, while other nodes keep far away from the query node.

**Step 3, identification target community**. Finally, all edges in the core edge set will converge, and the target community can be easily identified by removing the edges with distance equivalent to 1.

(5) *Algorithem K-Hop*

Let us analyze the time complexity of the K-Hop algorithm. For the first phase, the time complexity is $O(|E|)$, where $|E|$ is the number of all edges in the original large-scale network. For the second phase, due to the k-hop sub-network executing the local distance dynamic model, the time complexity is from the local dynamic interaction. In the local dynamic interaction, due to the initial distance of each edge being computed in the first phase, the time complexity is $O(s\cdot(|E'|)+T\cdot(|E'|)$, where $s$ is the average number of exclusive nodes and $T$ is the time steps ($0<T\leq 20$), and $|E'|$ is the number of edges in k-hop subgraph. For the three phases, all long edges can be removed in the second phase, so the time consumption of this phase is also saved. In summary, the whole time complexity of our proposed algorithm is $O(|E|)+s\cdot(|E'|)+T\cdot(|E'|))$.

## 5. Experiments

In this section, we perform extensive experiments to evaluate the effectiveness and efficiency of the proposed algorithm using a variety of real-world and synthetic networks.

### 5.1 Experiment Setup

(1) *Platform*

We implemented all algorithms in Python and ran the experiments on a Windows Server with 2 Intel Xeon E5-2600 series processors and 176GB main memory.

(2) *Networks*

We used six real-world large networks in our experiments: Amazon, DBLP, Youtube, LiveJournal, Orkut and Friendster. For each network, we only consider the target community that the query node belongs to. The statistics of these networks are shown in **Table 1**. These networks are provided with ground-truth community memberships and are publicly available at *https://snap.stanford.edu/data*.

**Table 1.** Real-world Networks.

| Network | #Node | #Edge | #Communities | Network type |
|---|---|---|---|---|
| Amazon | 334,863 | 925,872 | 75,149 | Undirected |
| DBLP | 317,080 | 1,049,866 | 13,477 | Undirected |
| Youtube | 1,134,890 | 2,987,624 | 8,385 | Undirected |
| LiveJournal | 3,997,962 | 34,681,189 | 287,512 | Undirected |
| Orkut | 3,072,441 | 117,185,083 | 6,288,363 | Undirected |
| Friendster | 65,608,366 | 1,806,067,135 | 957,154 | Undirected |

(3) *Comparison Algorithms*

We compare our K-Hop algorithm with two state-of-the-art community search algorithms, which are summarized in **Table 2**. All of the algorithms can be categorized into two classes: metric-based and natural-way. The k-core and k-truss algorithms belong to the first class and are required to optimize both internal denseness and external sparseness. The K-Hop algorithm belongs to the second class; in addition to optimizing both the internal denseness and external sparseness, it also needs to optimize the sharpness of the community boundary. For all experiments, without further statement, k-core and k-truss specify the default value of $k$ to 6.

**Table 2.** Comparison Algorithms.

| Algorithm | Full Name | Implement |
|---|---|---|
| K-Core [9] | Local search of communities in large graphs. | Python |
| K-Truss [11] | Querying k-truss community in large and dynamic graphs. | Python |
| K-Hop | K-hop community search based on local distance dynamics. | Python |

(4) *Evaluation Metrics*

To extensively compare different community search algorithms with respect to effectiveness, we adopt the following three quality measures for effectiveness testing.

*Relative Density*: The density is a popular community search fitness measure that takes both the inter and intra edges of a target community into consideration [10, 12, 13]. Intuitively, a target community with high density indicates that the connectivity of the target community is high. The density of the target community is formally defined as follows, where $C$ is the detected community, |*| indicates the number of * in the set, $E|C|$ is the number of inter edges, and $E'|C|$ is the number of intra edges.

$$density(C) = \frac{|E(C)|}{|E(C)| + |E'(C)|} \tag{11}$$

*Diameter*: The diameter of a target community has been considered an important feature of a community [10, 11, 28]. The diameter of a target community is the longest distance of all pairs of nodes in the community, where the distance is the minimum number of hops one node needs to reach another node.

*F-socre*: Intuitively, given the ground-truth community $C[T]$ and the detected community $C[D]$, we should find the community that resembles as much as possible the ground-truth community [29, 30]. *F-score* can measure the accuracy of the detected community with regard to the ground-truth community labels. The *F-score* is formally defined as follows, where $prec(D,T)$ is the precision and $rec(D,T)$ is the recall.

$$F(D,T) = 2 \cdot \frac{|prec(D,T) \cdot rec(D,T)|}{|prec(D,T) + rec(D,T)|} \tag{12}$$

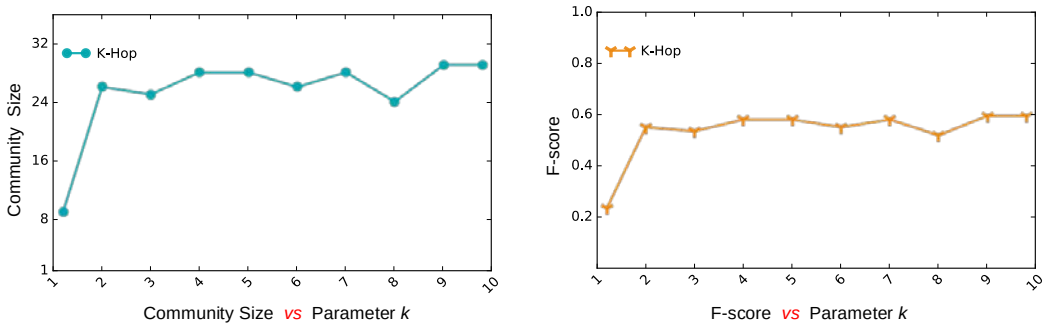
## 5.2 Influence of Parameters

K-Hop algorithm uses two parameters: $k$ and $\lambda$. In this subsection, we investigate the influence of $k$ and $\lambda$ on the result of the community. We are interested in the changes in community size and the accuracies with the different values of $k$ and $\lambda$.

(1) *Parameter k*

We first provide analysis on the influence of $k$ on community size and accuracies. For the K-Hop algorithm, the parameter $k$ is used to determine the scope of local interaction. In general, it is expected that the community size depends on small changes to $k$. When $k$ is small, the scope for a group of vertices forming a community is narrow. Consequently, it costs little time to find the community, but the value of the community size and F-score are small. When $k$ grows, the range of local interaction will become larger, and leads to a bigger search space. As result, it costs more time to find the community, but the value of community size and F-score are almost stable. To verify this conjecture, we studied the sensitivity of parameter $k$ in an LFR network. Similar results were obtained on other networks.

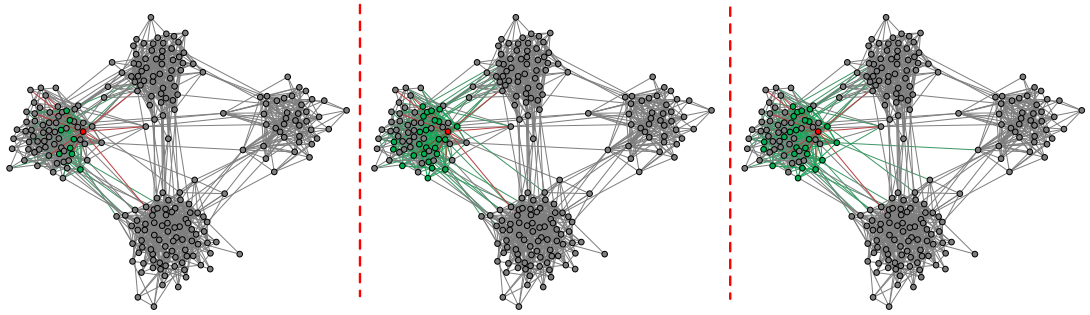
The results are shown in **Fig. 7**, and they verify our conjecture. From **Fig. 7(a)**, we can clearly see that the community size is very small when $k = 1$. When $k = 2$ and larger, the community size is almost stable. From **Fig. 7(b)**, we can clearly see that $k = 1$ is the critical point on which the minimum F-score is found. After this, the F-score is almost stable.




( a ) Community Size ( b ) F-score

**Fig. 7.** Sensitivity of parameter k.

**Fig. 8** plots the finding community with different $k$ ranging from 1 to 3 on a synthetic network (red nodes denote query nodes, and green nodes denote the membership of the finding community). From **Fig. 8(a) – 8(c)**, we can see that K-Hop always yields better results with parameter $k > 1$. In general, a $k$ value between 2 and 3 is normally sufficient to achieve a good clustering result.



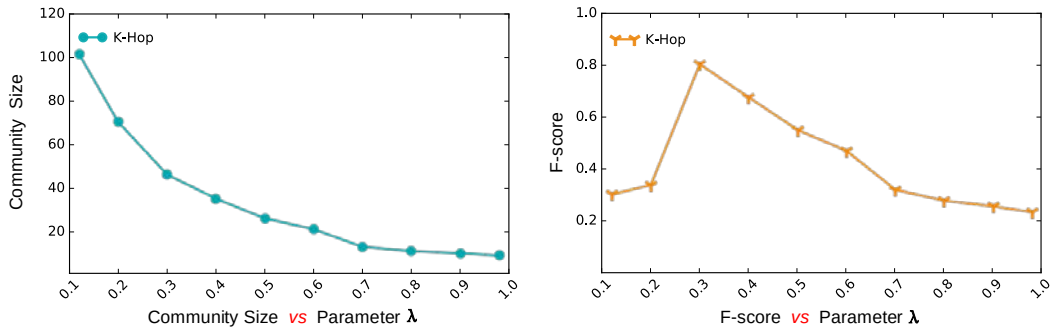
(a) k=1 (b) k=2 (c) k=3

**Fig. 8.** The sensitivity of parameter k on community search.

(2) *Parameter λ*

The value of λ is the most important parameter for the local distance dynamics model. For the K-Kop algorithm, the λ is used to determine the negative or positive interaction influence on the distances from exclusive neighbors. In general, it is expected that the community size monotonically decreases with λ. When λ is small, it is easy to find a large community. When λ grows, the community becomes smaller.

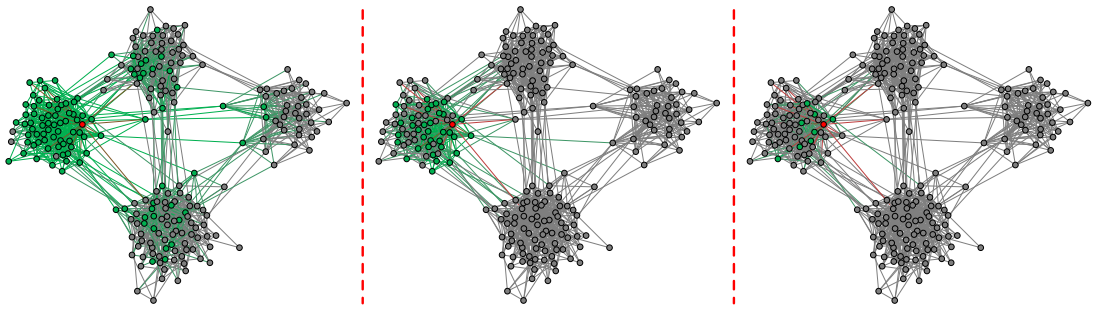
**Fig. 9** shows our results on a synthetic network. From **Fig. 9(a)**, we can clearly see that the community size monotonically decreases with λ. From **Fig. 9(b)**, we can clearly see that λ = 0.3 is the critical point on which the maximal value of the F-score is found. Before this, most nodes in the k-hop subgraph move together to the query node when λ increases. After this, irrelevant nodes quickly keep far away from the query node due to the strong constraint on the closeness of a community.




( a ) Community Size ( b ) F-score

**Fig. 9.** Sensitivity of parameter λ.

**Fig. 10** plots the finding community with different λ ranging from 0 to 1 on a synthetic network (red nodes denote query nodes, and green nodes denote the membership of the finding community). From **Fig. 10(a) – 10(c),** we can see that K-Hop always yields better results with parameter $\lambda > 0.3$. In general, a λ value between 0.3 and 0.6 is normally sufficient to achieve a good clustering result.



(a) λ=0.1 (b) λ=0.5 (c) λ=1.0

**Fig. 10.** The sensitivity of parameter λ on community search.

From the results, we found that K-Hop is not sensitive to the search results. In general, a λ value between 0.3 and 0.6 is normally sufficient to achieve a good result. We recommend a value of 2 for *k*. For K-Hop, we set $k = 2$ and $\lambda = 0.5$ as default parameters.

## 5.3 Evaluation on Real Networks

(1) *Effectiveness Evaluation*

We first evaluate the effectiveness of the selected methods on real networks. For each network, we randomly select 100 query nodes with degree ranging from 10 to 100. The query node is selected from a random ground-truth community.

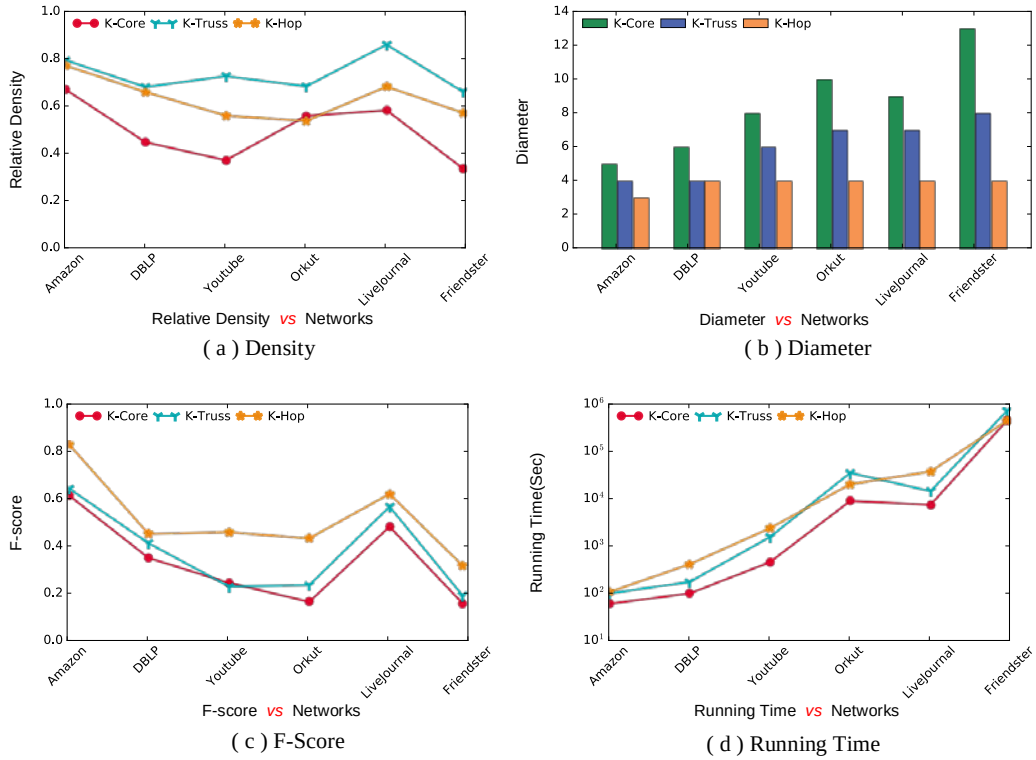



**Fig. 11.** Evaluation on real networks.

**Fig. 11(a)** shows the relative density of the selected algorithms on different networks. It can be seen that the K-Truss method is better than other methods on most networks. Focusing on the K-Core and K-Hop algorithms, it is not difficult to find that the performance of the K-Hop algorithm exceeds that of the K-Core method. In addition, the K-Hop algorithm is very close to the K-Truss algorithm on some real-world networks.

**Fig. 11(b)** discusses the diameter of the community search of various algorithms on real-world networks. From **Fig. 9**, we can get the following observations. (1) For K-Core, the value of the diameters on six networks are very uneven, which implies that the performance of the K-Core algorithm is very unstable on real-world networks. (2) For K-Truss, we find that K-Truss has better results and stability than the K-Core algorithm on real-world networks. (3) For K-Hop, six real-world networks have good results, and the average value of the diameter is less than 4. (4) Focusing on K-Core and K-Truss, we observe that these algorithms have larger diameters, and this may be caused by the free rider effect.

**Fig. 11(c)** shows the F-score of the identified community using different algorithms. We can see that the F-score value of K-Hop is 5% to 10% higher than those of other methods. If the nodes in the irrelevant community are selected as the identified community, these algorithms will identify irrelevant communities and will cause a low F-score value.

(2) *Efficiency Evaluation*

In this section, we compare the overall running time of different methods. **Fig. 11(d)** shows the overall running time averaged over 100 random query nodes. From **Fig. 11(d)**, when the scale of the network is small, the running time of all algorithms is small. Along with the increase in scale of the network, the running time increases gradually. It is important to note that the K-Core runs faster than K-Truss and K-Hop, but its accuracy is low. The K-Truss and K-Hop algorithms have similar performance.

### 5.4 Evaluation on Synthetic Networks

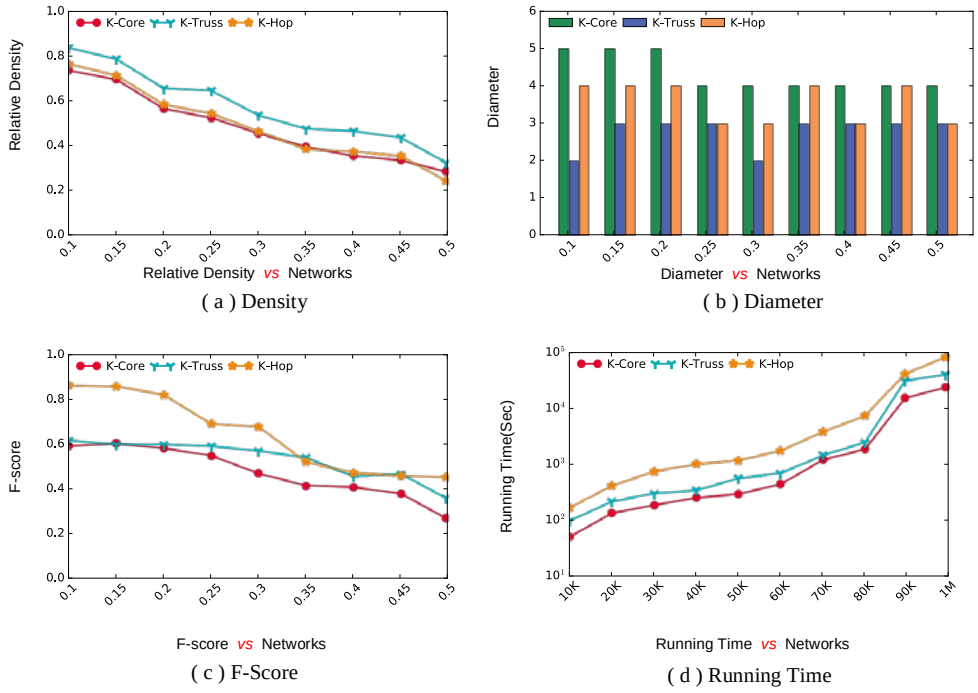
To test the sensitivity to community structure and the scalability of our method, the LFR benchmark is selected to generate multiple synthetic networks in this paper. The LFR benchmark is a baseline network and has been considered as a *de facto standard model*; it was proposed by Lancichinett [22] for generating networks that overcome the drawbacks of the GN benchmark [15]. In this paper, three important parameters of the LFR benchmark are selected as basic parameters for generating different synthetic networks. **Table 3** shows our selected parameters for the benchmark in detail. The mixing parameter $\mu$ indicates the proportion of a node's neighbors that belong to other communities. The larger the value of $\mu$, the less clear the community structure is.

**Table 3.** Parameters for the LFR benchmark (our selected).

| Parameter | Description |
|---|---|
| $N$ | The number of nodes. |
| $k$ | The average degree. |
| $\mu$ | The mixing parameter. |

We first evaluate the sensitivity to community structure. In general, the more clear the community structure is, the more efficient the community search. To verify this conjecture, we generate synthetic networks by varying the mixing parameter μ from 0.1 to 0.5. The number of nodes in the network is 100K, and the average degree is 10.

We present the result of the relative density in **Fig. 12(a)**. As we can see, the relative density decreases when increasing the mixing parameter $u$. The K-Truss method has high relative density and slightly outperforms other algorithms. Moreover, the K-Hop algorithm is very robust to the mixing parameter $u$ with a slight drop in relative density value for larger $u$ values.




( a ) Density ( b ) Diameter

( c ) F-Score ( d ) Running Time

**Fig. 12.** Evaluation on synthetic networks.

**Fig. 12(b)** discusses the diameter of the community search of various algorithms on LFR synthetic networks. Comparing three algorithms, we can find that K-Truss and K-Hop have the better value and stability. The K-Core algorithm is worst. For K-Core, the value of the diameters on different mixing parameters $u$ are very uneven, which implies that the performance of the K-Core algorithm is very unstable on synthetic networks.

**Fig. 12(c)** shows the F-score of three algorithms on LFR synthetic networks. From **Fig. 12(c)**, we find that the F-score of the K-Hop algorithm is much larger than that of other algorithms and the accuracy of K-Hop is best. The K-Core and K-Truss are very close, and the accuracy of K-Truss is next. The F-score of the K-Core algorithm is much smaller than that of K-Hop, and the accuracy is the worst. Moreover, the F-score of all algorithms drops significantly once the mixing parameter $u$ is larger than 0.3; the boundary between the communities becomes less clear with the increase in value of $u$. It is easier for all methods to select the irrelevant nodes of the target community. As a result, the value of the F-score becomes smaller.

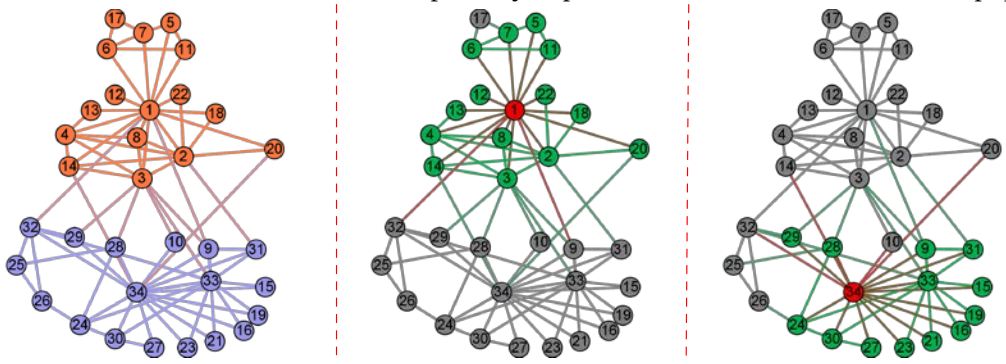
We then evaluate the scalability by using the above network model. We generate an ensemble of synthetic networks by varying the number of nodes $N$ from $100K$ to $1M$. The average degree is 10, and the mixing parameter is 0.3.

The scalability results of the community search are shown in **Fig. 12(d)**. From **Fig. 12(d)**, we find that along with the increase in scale of the network, the running time increases gradually. The local community search algorithm K-Core has nearly the lowest running time. The K-Truss algorithm also uses a local search procedure but has to compute the support of each edge on the whole graph; thus, it has a much longer running time than K-Core. Although the K-Core and the K-Truss are somewhat faster than K-Hop, they suffer in terms of the

quality of the resulting community search.

### 5.5 Case Study

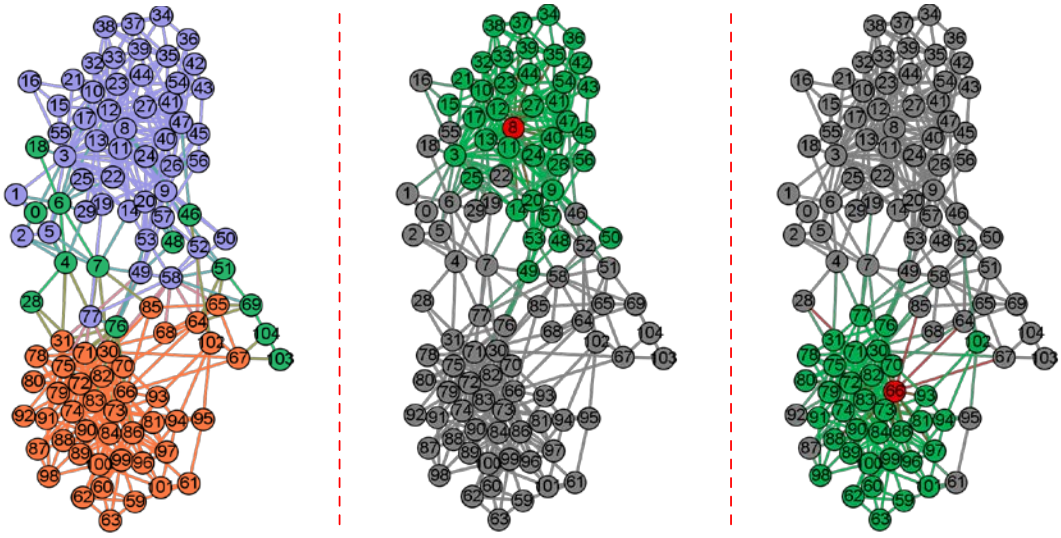
To validate the effectiveness of our local distance dynamics model, we select two well-known real-world networks with the ground truth, using the K-Hop algorithm to search the community structure with different query nodes. Three real-world networks are publicly available from the UCI network data repository *https://networkdata.ics.uci.edu/index.php*.



**( a ) ground truth (2 classes, 34 nodes, and 78 edges)** **( b ) the community identified by K-Hop with 1 as the query node** **( c ) the community identified by K-Hop with 34 as the query node**

**Fig. 13.** Case study on Zachary's karate club.

The first network is the Zachary's karate club network, consisting of 34 vertices and 78 undirected edges. Each node represents a member of the club, and each edge represents a tie between two members of the club. In this case study, we use node "1" and "34" as the query node. After setting $k = 2$ and $\lambda = 0.5$, we obtained the community result shown in **Fig. 13**. **Fig. 13(a)** shows the ground truth of the karate club network, which covers 2 classes. **Fig. 13(b)** shows the detection results with "1" as the query node, denoted by the green nodes. **Fig. 13(c)** shows the detection results with "34" as the query node, denoted by the green nodes.



**( a ) ground truth( 3 classes, 105 nodes, and 441 edges )** **( b ) the community identified by K-Hop with 8 as the query node** **( c ) the community identified by K-Hop with 66 as the query node**

**Fig. 14.** Case study on Books about US politics.

The second network is the Books about US politics network, consisting of 105 nodes and 441 edges. The network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller "Amazon.com". Here, we use node "8" and "66" as the query node. After setting $k$ = 2 and $\lambda$ = 0.5, we obtained the community result shown in **Fig. 14**. **Fig. 14(a)** shows the ground truth of the network, covering 3 classes. **Fig. 14(b)** shows the detection results with "8" as the query node, denoted by the green nodes. **Fig. 14(c)** shows the detection results with "66" as the query node, denoted by the green nodes.

Based on the above two cases studies, we make the following remarks. (1) Our local distance dynamics model is the effectiveness of different networks. (2) K-Hop algorithm can accurately search the community structure of different networks with different query nodes.

## 6. Conclusion

Most existing community search algorithms optimize a goodness metric and usually suffer from the free rider effect. In this paper, we introduce a new community search algorithm, called K-Hop, to automatically find the best community containing a query node in networks based on local distance dynamics. Extensive experiments are executed on the synthetic network as well as the real-world network. Compared with several state-of-the-art methods, the experimental results show the attractive benefits of our algorithm. Our future work will consider the community search on heterogeneous networks based on the intuitive local distance dynamic model.

## Acknowledgement

This work was supported by the National Natural Science Foundation of China (61573299,61174140, 61472127, 61272395); Social Science Foundation of Hunan Province(16ZDA07); China Postdoctoral Science Foundation (2013M540628,2014T70767); Natural Science Foundation of Hunan Province (14JJ3107); Excellent Youth Scholars Project of Hunan Province (15B087).

## References

[1] Xu X, Yuruk N and Feng Z, "Scan: a structural clustering algorithm for networks," in *Proc. of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.824-833, August, 12-15, 2007. Article (CrossRef Link).

[2] Newman M E J, "Modularity and community structure in networks," *Proceedings of the national academy of sciences*, vol.103, no.23, pp.8577-8582, June, 2006. Article (CrossRef Link).

[3] Shao J, Han Z, Yang Q and Zhou T, "Community detection based on distance dynamics," in *Proc. of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.1075-1084, August, 10-13, 2015. Article (CrossRef Link).

[4] Raghavan U N, Albert R and Kumara S, "Near linear time algorithm to detect community structures in large-scale networks," *Physical review E*, vol.76, no.3, pp.36-106, September, 2007. Article (CrossRef Link).

[5] Newman M E J and Girvan M, "Finding and evaluating community structure in networks," *Physical review E*, vol.69, no.2, pp.26-113, February, 2004. Article (CrossRef Link).

[6] Sozio M and Gionis A, "The community-search problem and how to plan a successful cocktail party," in *Proc. of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.939-948, July, 25-28, 2010. Article (CrossRef Link).

[7] Ugander J, Backstrom L and Marlow C, "Structural diversity in social contagion," *Proceedings of the National Academy of Sciences*, vol.109, no.16, pp.5962-5966, April, 2012. Article (CrossRef Link).
[8] Zhang S, Liu Q and Lin Y, "Anonymizing popularity in online social networks with full utility," *Future Generation Computer Systems*, vol.72, no.1, pp.227-238, July, 2017. Article (CrossRef Link).
[9] Cui W, Xiao Y, Wang H and Wang W, "Local search of communities in large graphs," in *Proc. of the 2014 ACM SIGMOD International Conference on Management of Data*, pp.991-1002, June, 22-27, 2014. Article (CrossRef Link).
[10] Li R H, Qin L and Yu J X, "Influential community search in large networks," *VLDB Endowment*, vol.8, no.5, pp.509-520, January, 2015. Article (CrossRef Link).
[11] Huang X, Cheng H, Qin L and Tian W, "Querying k-truss community in large and dynamic graphs," in *Proc. of the 2014 ACM SIGMOD International Conference on Management of Data*, pp.1311-1322, June, 22-27, 2014. Article (CrossRef Link).
[12] Huang X, Lakshmanan L V S and Yu J X, "Approximate closest community search in networks," *VLDB Endowment*, vol.9, no.4, pp.276-287, December, 2015. Article (CrossRef Link).
[13] Wu Y, Jin R, Li J and Zang X, "Robust local community detection: on free rider effect and its elimination," *VLDB Endowment*, vol.8, no.7, pp.798-809, February, 2015. Article (CrossRef Link).
[14] Xie J, Kelley S and Szymanski B K, "Overlapping community detection in networks: The state-of-the-art and comparative study," *ACM Computing Surveys (CSUR)*, vol.45, no.43, pp.1-35, August, 2013. Article (CrossRef Link).
[15] Palla G, Derényi I and Farkas I, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol.435, no.7043, pp.814-818, June, 2005. Article (CrossRef Link).
[16] Lancichinetti A, Fortunato S and Kertész J, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, vol.11, no.3, pp.15-33, March, 2009. Article (CrossRef Link).
[17] Fortunato S, "Community detection in graphs," *Physics reports*, vol.486, no.3, pp.75-174, February, 2010. Article (CrossRef Link).
[18] Huang J, Sun H, and Liu Y, "Towards online multiresolution community detection in large-scale networks," *PloS one*, vol.6, no.8, pp. e23829, August, 2011. Article (CrossRef Link).
[19] Cheng J, Zhu L and Ke Y, "Fast algorithms for maximal clique enumeration with limited memory," in *Proc. of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.1240-1248, August, 12-16, 2012. Article (CrossRef Link).
[20] Zhou X, Li K and Xiao G, "Top k Favorite Probabilistic Products Queries," *IEEE Transactions on Knowledge and Data Engineering*, vol.28, no.10, pp.2808-2821, June, 2016. Article (CrossRef Link).
[21] Wang J and Cheng J, "Truss decomposition in massive networks," *VLDB Endowment*, vol.5, no.9, pp.812-823, May, 2012. Article (CrossRef Link).
[22] Chang L, Yu J X and Qin L, "Efficiently computing k-edge connected components via graph decomposition," in *Proc. of the 2013 ACM SIGMOD International Conference on Management of Data*, pp.205-216, June, 2013. Article (CrossRef Link).
[23] Barahona M and Pecora L M, "Synchronization in small-world systems," *Physical review letters*, vol.89, no.5, pp.54-101, July, 2002. Article (CrossRef Link).
[24] Watts D J and Strogatz S H, "Collective dynamics of 'small-world' networks," *nature*, vol.393, no.6684, pp.440-442, June, 1998. Article (CrossRef Link).
[25] Liu Q, Wang G, and Li F, "Preserving privacy with probabilistic indistinguishability in weighted social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol.28, no.5, pp.1417-1429, May, 2017. Article (CrossRef Link).
[26] Kunze M, Weidlich M and Weske M, "Behavioral similarity–a proper metric," in *Proc. of the 2011 Springer Berlin Heidelberg International Conference on Business Process Management*, pp.166-181, August, 2011. Article (CrossRef Link).

[27] Lipkus A H, "A proof of the triangle inequality for the Tanimoto distance," *Journal of Mathematical Chemistry*, vol.26, no.3, pp.263-265, October, 1999. Article (CrossRef Link).
[28] Edachery J, Sen A and Brandenburg F J, "Graph clustering using distance-k cliques," in *Proc. of the 7th International Symposium on Graph Drawing*, pp.98-106, March, 1999. Article (CrossRef Link).
[29] Xiao G, Li K, and Li K, "Efficient top-(k, l) range query processing for uncertain data based on multicore architectures," *Distributed and Parallel Databases*, vol.33, no.3, pp.381-413, October, 2015. Article (CrossRef Link).
[30] Luo J and Wang T, "Motif discovery using an immune genetic algorithm," *Journal of theoretical biology*, vol.264, no.2, pp.319-325, May, 2010. Article (CrossRef Link).

**Tao Meng** received the M.S. degree in College of Information Science and Engineering from Hu Nan University, Changsha, China. He is currently pursuing the Ph.D. degree in the College of Information Science and Engineering, Hu Nan University, Changsha, China. His research interests include date mining, pattern recognition.

**LiJun Cai** received the Ph.D. degree in College of Information Science and Engineering from Hu Nan University in 2007. He is currently a Professor at Hu Nan University. His research interests include bioinformatics, cloud computing, big data scheduling and management.

**TingQin He** received the M.S. degree in College of Information Science and Engineering from Hu Nan University, Changsha, China. He is currently pursuing the Ph.D. degree in the College of Information Science and Engineering, Hu Nan University, Changsha, China. His research interests include date mining, pattern recognition.

**Lei Chen** received the M.S. degree in College of Information Science and Engineering from Hu Nan University, Changsha, China. He is currently pursuing the Ph.D. degree in the College of Electrical and Information Engineering, Hu Nan University, Changsha, China. His research interests include date mining, cloud computing, big data scheduling and analysis.

**ZiYun Deng** received the Ph.D. degree in College of Electrical and Information Engineering from Hu Nan University in 2016. He is currently a Professor at ChangSha Commerce Tourism College. His research interests include high performance computing, logistics information technology.