International Journal of Innovative
Computing, Information and Control
Volume **16**, Number **4**, August **2020**

ICIC International ©2020 ISSN 1349-4198
pp. **1183–1201**

GROUP CENTRALITY ALGORITHMS BASED ON THE H-INDEX FOR IDENTIFYING INFLUENTIAL NODES IN LARGE-SCALE NETWORKS

Yixin Li[1,2], Yiqiang Sheng[1,2,*] and Xiaozhou Ye[1,2]

[1]National Network New Media Engineering Research Center
Institute of Acoustics, Chinese Academy of Sciences
No. 21, North 4th Ring Road, Haidian District, Beijing 100190, P. R. China
{ liyixin; yexz }@dsp.ac.cn; *Corresponding author: shengyq@dsp.ac.cn

[2]School of Electronic, Electrical and Communication Engineering
University of Chinese Academy of Sciences
No. 19(A), Yuquan Road, Shijingshan District, Beijing 100049, P. R. China

Received February 2020; revised May 2020

Abstract. *Our target is to investigate influence maximization because it has numerous applications, such as using influence maximization to spread ideas in social networks, or inversely, avoiding influence maximization in order to control sources of infection in a population. Identifying influential nodes in complex networks is a fundamental task in influence maximization. The main challenges include scalability, dynamics, and topological diversity. By focusing on the scalability of spreaders with large-scale datasets, we extend the variant h-index to one of group centralities and investigate it as a new centrality, called the h-index group centrality, which fuses the h-index with the group. Furthermore, we propose five algorithms based on the h-index group centrality. These five algorithms satisfy the requirements of large-scale networks. Experimental results show that our algorithms outperform comparison algorithms and that our algorithms have great scalability on tested datasets. In the best case, the results of our method exceed those of the latest method by more than 1,000 times. Group centrality algorithms based on the h-index achieve significant improvement and have many potential applications.*
Keywords: Influence maximization, Large-scale networks, H-index, Group centrality

1. **Introduction.** Identifying influential nodes has become a necessity in many practical fields. As the first example, in the field of preventive medicine, identifying the most vulnerable people helps effectively prevent infectious diseases. Second, finding the most influential nodes in a network topology helps lay a foundation for ensuring the security and reliability of the network [1]. Third, if enterprises collaborate with the most influential media for advertising, they will produce the best commercial results [2-4]. Fourth, in the field of social network analysis [5] that has attracted attention in recent years, after defining the relationship among actors, network analysis can be carried out with the help of the study of influence maximization.

The key to these examples is to identify the most influential nodes in a network. There have been previous studies [6,7] in the field of influence maximization. Richardson and Domingos [8] defined the problem of influence maximization as an algorithmic problem in social networks. Kempe et al. [9] proposed a greedy strategy to solve the problem. Other than that, many heuristic methods [10,13,14] have been proposed. Among the heuristics, the centrality algorithms, such as degree, betweenness, and closeness, are very representative. In addition, excellent algorithms such as PageRank, its related algorithms

DOI: 10.24507/ijicic.16.04.1183

[11,12], and the core covering algorithm [13] have been published to solve the influence maximization problem. The objective of the paper is to investigate a useful new centrality to identify influential nodes.

There exist some challenges in identifying influential nodes. On the one hand, the accuracy of former heuristic methods is not satisfactory [15]. On the other hand, the time complexity of greedy algorithms is too high to be scalable [13]. To overcome these challenges, there are two main directions. The first direction focuses on the greedy strategy, and the main challenge is to reduce the time complexity. Although many improved algorithms [16,17] have been proposed, the time cost is not low enough to overcome the challenges. The second direction focuses on the heuristic strategy, and the main challenge is to improve the accuracy. In the previous research, this direction was proven to be feasible [14]. The optimization problem of identifying the most influential nodes is NP-hard [9]. Considering the practical applications for large-scale networks and the research value of an NP-hard problem, we chose the second direction and investigated a new centrality, called the h-index group centrality.

The degree and k-core are widely used in influence maximization. It has already been proven that the h-index is a better method than the degree and k-core for quantifying node influence in many cases [18]. Based on the group concept in set theory and the concept of a variant h-index, we investigate a new centrality method called the h-index group centrality. We hope to use the h-index group centrality as a better method to quantify the influence of nodes in large-scale networks. We proposed a series of algorithms based on the h-index group centrality and carried out hundreds of simulation experiments using the independent cascade (IC) model [9,19] over four datasets to evaluate the h-index group centrality.

Our major contributions are as follows.

1) We extend the variant h-index to one of the group centralities in order to more accurately identify influential nodes. To the best of our knowledge, this is the first time that the h-index group centrality has been investigated. Additionally, we extend the degree centrality into the degree group centrality with experimental evaluations.
2) We propose five group centrality algorithms. Three of them are trivial group algorithms, which are fusions of the node and the variant h-index. The other two algorithms are community group algorithms, which form groups through community detection and fuse the community with the variant h-index. The first type of our algorithms includes the h-index trivial group algorithm (HTGA), h-index trivial group with distance algorithm (HTGDA), and h-index trivial group deleting intersection algorithm (HTGDIA). The other type includes the h-index community group algorithm (HCGA) and h-index community group summing algorithm (HCGSA).
3) Our experimental results show that h-index group centrality and its algorithms offer significant effectiveness. What deserves to be mentioned is that HCGSA offers greatly scalable performance with respect to accuracy for some datasets. In the best case, the experimental results of HCGSA exceed the experimental results of the latest algorithm by 1205.3 times.

The rest of this paper is organized as follows. Section 2 describes the previous knowledge. Section 3 presents the formulation, algorithms, and experimental steps. Section 4 provides the materials, results, and discussion. Section 5 provides conclusion.

2. **Previous Knowledge.**

2.1. **Degree centrality.**

The number of neighbor nodes of a node is called the degree [10] of the node.

Formula (1) calculates the degree centrality.

$$C_d(v_i) = \sum_{j=1}^{n} x_{ij}, \quad i \neq j, \tag{1}$$

where x_{ij} takes the value of either "0" or "1". "1" means that there is an edge between node v_i and node v_j. "0" means that node v_i and node v_j are not directly connected.

The degree centrality is the most convenient method to measure the influence of nodes, and it is suitable for combination with other methods [20].

2.2. ***k*-core.** The concept of the k-core [21] was proposed by Seidman in 1983. The k-core is used to describe the network characteristics that the degree cannot describe, such as structural characteristics and hierarchical characteristics.

If we recursively remove the nodes whose degrees are less than k, along with all edges connected to these nodes, then we will obtain a subgraph. The subgraph is the k-core of the whole graph. If node v belongs to the k-core and does not belong to the $k+1$-core, then k will be called the coreness of node v. In k-core-based algorithms [22], the larger coreness a node has, the more influential the node is.

An algorithm based on the k-core method was proposed by Cao et al. in 2015. It is called the coreness covering algorithm (CCA) [13], and it aims to increase the final infection rate. The central idea of CCA is to use the k-core and distance to select the set of spreaders [15,23]. The n-hop neighbor nodes of node A will all be updated to the covering state when node A is selected as a spreader. Covered nodes are not allowed to be selected as spreaders.

2.3. **H-index.** The h-index [24] was proposed by Hirsch in 2005 and was defined as the number of papers with citation number no less than h. The h-index is an effective method to quantify the scientific output of a researcher. Additionally, the h-index has many applications [2,25,26] in other fields.

2.4. **Community detection.** Studies about the structural and topological characteristics of networks are useful tools [27,28] for identifying influential nodes.

The Louvain algorithm [30] is a common algorithm for community detection [29] in large-scale networks. The Louvain algorithm was proposed by Blondel et al. in 2008. It has undergone some extension studies [31,32].

In the Louvain algorithm, the quality of community detection can be measured by modularity Q [30], which is calculated by Formula (2) [30].

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{2}$$

where A_{ij} represents the weight of the edge between node i and node j, $k_i = \sum_j A_{ij}$ is the sum of the weights of the edges connected to node i, c_i is the community to which node i is assigned, the function δ is 1 if $c_i = c_j$ and 0 otherwise, and $m = \frac{1}{2} \sum_{ij} A_{ij}$.

This algorithm can be roughly summarized into four steps [30]. First, we assign every node to a community. Second, we change the community of node i that has a neighbor named node j. We move i to the community where node j is assigned. Third, we calculate the increment ΔQ. If $\Delta Q > 0$, then we will apply this modification. We then return to step two and loop the process with other nodes until modularity Q does not grow or other termination conditions are attained. Fourth, we use the results in step three to rebuild the graph. After rebuilding the graph, we loop steps two and three. This process will stop when the structure of the graph is stable.

2.5. **Propagation.** The IC model is a basic model for spreading [33,34]. Its main idea can be divided into four steps. First, a set of the most influential nodes (i.e., spreader set) should be obtained. Second, we use node u in the spreader set to activate its neighbor node v that is not in the spreader set. The probability of success for propagation from u to v is p_{uv}. Third, if node v is activated by u, then v will be able to affect its own neighbor nodes. Fourth, we loop this process under certain constraints until no more new nodes are activated in a single loop or any other termination condition is reached.

2.6. **ClusterRank.** To identify influential nodes, Wang et al. proposed a local clustering coefficient algorithm, ClusterRank [20], in 2017. Although IO-ClusterRank is similar to PageRank, LeaderRank, betweenness and closeness in terms of node exploitation for networks with uniform out/in-degree distribution, in most other networks, it mines nodes with more influential ability and has a faster propagation speed [20].

3. Methodology.

3.1. **Formulation.** To investigate the group centrality algorithms, we introduce the concept of a group. A group can be regarded as a set of selected nodes in a network.

A random group is a group formed through random sampling. For example, in psychological experiments, experimental subjects with the same properties except for the independent variable are usually grouped by random sampling. A trivial group is a group formed by a single element. For example, in air traffic control, every aircraft can be regarded as a trivial group. A community is a group formed through community detection. For example, on social networking sites, users who are closely related might be divided into the same virtual community.

3.1.1. *Group centrality based on degree.* According to Formula (1), we define degree group centrality $C_d(S_i)$ for group S_i as Formula (3).

$$C_d(S_i) = \sum_{j=j_1}^{j_{n_i}} \sum_{k=1}^{n} x_{jk}, \quad j \neq k,\ v_k \notin S_i, \tag{3}$$

where x_{jk} is "1" when there is an edge between v_j and v_k and "0" otherwise, $i \in [1, m]$ (m is the number of groups in the network), $i \in \mathbb{Z}$, n_i is the number of nodes in S_i, n is the number of nodes in the entire network, j_1 is the first subscript in S_i, and j_{n_i} is the last subscript in S_i.

By investigating the concept of a group, we find that "node" and "community" are two special forms of "group". If we divide a network into extremely small groups, where there is only one node in each group, then the degree group centrality will be equal to the original degree centrality. If we divide a network through community detection, then the groups will be communities, and the degree group centrality will be the degree community centrality.

3.1.2. *Group centrality based on the h-index.* We use an operator $\mathcal{H}$ from [18] that takes a descending sequence $\{x_1, x_2, \ldots, x_n\}$ as the independent variable to obtain an integer y, $y = \mathcal{H}(x_1, x_2, \ldots, x_n) > 0$. The dependent variable y is the maximum integer satisfying the $y \leq x_y$ condition.

Applying this definition to a network, x_i is the degree of node v_i, and y is the h-index centrality of node v_i. The definition of the h-index centrality for a node is as given in Formula (4).

$$C_h(v_i) = \mathcal{H}(d_1, \ldots, d_{n_i}), \quad v_i \in V, \tag{4}$$

where V is the complete set of all nodes in the network, n_i is the number of neighbor nodes of v_i, d_j is the degree of v_j ($j \in [1, n_i]$, $j \in \mathbb{Z}$), and v_j is one of the neighbor nodes of v_i. A simple example is shown in Figure 1(b).

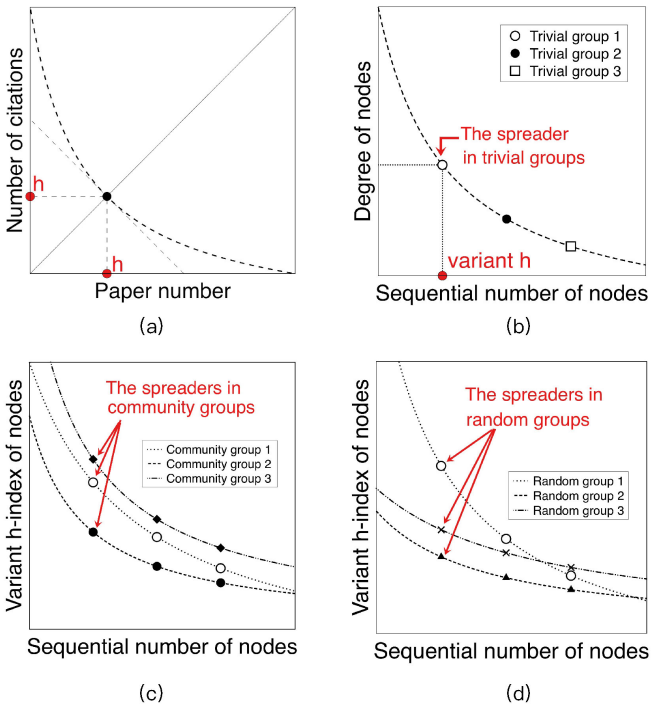


FIGURE 1. The original and variant concepts of the h-index. (a) presents the original definition of the h-index [24]; (b) presents the definition of the variant h-index to be extended to one of our group centralities by fusion with the concept of a group; (c) presents how we select spreaders according to the variant h-index in community groups; (d) presents how we select spreaders according to the variant h-index in random groups.

Thus, we can obtain the definition of the h-index group centrality as Formula (5):

$$C_h(S_i) = \sum_{j=1}^{m} \mathcal{H}\left(d'_1, \ldots, d'_{n_j}\right), \tag{5}$$

where m is the number of nodes in group S_i.

The difference between d_{n_j} and d'_{n_j} is that for d'_{n_j}, part of the degree is subtracted from d_{n_j}. The subtracted part represents the connections between v_j and the nodes in the same group as v_j.

The definition of the h-index group centrality can be extended to the h-index node centrality and the h-index community centrality, considering that "node" and "community" are special forms of "group".

After completing the definition of the h-index group centrality, we can use this definition to consider whether a specific node can join the spreader set. If node $\mathcal{F}(G_i)$ satisfies Formula (6), then node $\mathcal{F}(G_i)$ will be selected into the most influential node set.

$$\frac{\left|N^{(r)}(\mathcal{F}(S_i)) \cap N^{(n)}(SP)\right|}{|N^{(n)}(SP)|} \leq \mu, \tag{6}$$

where S_i is a group in network G, $\mathcal{F}$ is a function that returns a node $\mathcal{F}(S_i)$, node $\mathcal{F}(S_i)$ is the current best node in group S_i, $N^{(r)}(\mathcal{F}(S_i))$ is a set that contains 1-hop to r-hop

neighbor nodes of node $\mathcal{F}(S_i)$, SP is the spreader set, $N^{(n)}(SP)$ is a set that contains 1-hop to n-hop neighbor nodes of SP, and the threshold μ is a value used as a criterion for evaluating whether node $\mathcal{F}(S_i)$ can be selected into the spreader set.

If $\mu \in [1, +\infty)$, the inequality of Formula (6) is always true, regardless of the left side of Formula (6). In other words, the process of selecting the spreader set is only related to the h-index trivial group centrality or h-index community group centrality, and no other restrictions are imposed. This idea is reflected in our Algorithms 1, 4, and 5 in Section 3.2.

If $\mu = 0$, then the inequality of Formula (6) is true only when the left side of Formula (6) is equal to 0. That is, as long as there is any intersection between the $N^{(r)}(\mathcal{A})$ set of point $\mathcal{A}$ and the $N^{(n)}(SP)$ set of the spreader set, point $\mathcal{A}$ cannot be selected into the spreader set. In this case, the strictness of whether or not $\mathcal{A}$ can join the spreader set is determined by the values of r and n. This idea is reflected in our Algorithm 2 in Section 3.2.

If $\mu \in (0, 1)$, then whether $\mathcal{A}$ can join the spreader set depends on the values of r, n and μ. This idea is reflected in our Algorithm 3 in Section 3.2.

We wanted to find a formula to determine whether a node can join the spreader sets for different networks, and we wanted to propose an algorithm based on this formula. We have proposed such an inequality to evaluate nodes, which is Formula (6), but it is still a problem to propose only one algorithm for different networks. The reason is that we need to adjust r, n, μ and S_i in Formula (6) according to the characteristics of different networks. Some key adjustments affect whether the algorithm needs to modify some branches.

We have unified the evaluation of nodes on different networks with Formula (6) as much as possible, adjusted Formula (6) according to the characteristics of different networks, and then proposed algorithms for different forms of Formula (6).

We analyze a network in this way: in the plane coordinate system, the X axis corresponds to the h-index of the node, and the Y axis corresponds to the number of nodes. There is a point (m, n) in the coordinate system, indicating that there are n points with h-index value of m in the network.

After analyzing a network like this, we can produce a fitted function curve. Our goal is to find the most influential nodes. With the h-index as the criterion, we can easily find that such nodes are concentrated at the end of the curve.

Combined with such analysis, we will illustrate as follows which types of networks our algorithms fit. Assume that the size of the spreader set is n, the tail of the curve includes nodes whose sum of y-coordinates is not greater than n, and the selecting order is from the end of the curve to the front.

1) A long tail refers to a tail whose distance in the x-axis direction among the nodes is far. If the curve has a long tail whose y-coordinates of nodes are 1 or 2, then the spreader set can be directly selected by the h-index group centrality without interference from nodes that have the same h-index. Even if there are two nodes with the same h-index value, they can be distinguished by the degree and coreness. Algorithm 1 in Section 3.2 fits such networks.
2) If the curve has a tail in which some of the y-coordinates of its nodes are between 1 and 10, which means that there are multiple nodes with the same high h-index value, then it is not sufficient to select spreaders with the h-index group centrality. An algorithm with the step of "distinguishing the influence of nodes with the same high h-index" is required. Algorithms 2 and 3 in Section 3.2 fit such networks.

3) If the curve has a tail whose y-coordinates of nodes are much larger than the size of the spreader set, then a community discovery algorithm needs to be introduced to more effectively determine the influential nodes. Algorithm 5 in Section 3.2 is applicable to such networks, and Algorithm 4 in Section 3.2 is a transition algorithm of Algorithm 5.

We cannot claim that Formula (6) fits to all networks, but it fits several kinds of networks with which we have experimented, which we will describe in detail.

3.2. **Algorithms.** We apply the h-index group centrality to algorithm design. The calculation method of the h-index group centrality is as shown in Formula (7) and Formula (8).

Calculate the h-index of node v as follows.

1) Sort all neighbor nodes of node v in descending order:

$$\begin{aligned} &i, j \in \mathbb{Z}, \quad \forall i \in [1, n], \forall j \in [i+1, n], \\ &\text{s.t. } d_i \geq d_j, \end{aligned} \tag{7}$$

where n is the number of neighbor nodes of node v, d_i refers to the degree of the i-th neighbor node of node v, and d_j refers to the degree of the j-th neighbor node of node v.

2) Iteratively calculate the h-index until the h-index is stable:

$$h = \max[h, i \cdot bool(d_i \geq i)], \quad i \in [1, n], i \in \mathbb{Z}, \tag{8}$$

where $(d_i \geq i)$ is a Boolean value.

The following five h-index group centrality algorithms have a common and unique purpose: to find the most influential spreaders in the entire network. They are based on Formula (6), Formula (7) and Formula (8).

Algorithm 1 reflects a special form of the h-index group centrality when a group is a node. We refer to such groups that have single nodes as trivial groups. Formula (9) is a special form of Formula (6), and it is the theoretical basis of Algorithm 1.

$$\begin{aligned} &\exists \mu, \mu \in [1, +\infty), \\ &\text{s.t. } \frac{\left|N^{(r)}(\mathcal{F}(v_i)) \cap N^{(n)}(SP)\right|}{|N^{(n)}(SP)|} \leq \mu, \end{aligned} \tag{9}$$

where node v_i is the node that needs to be evaluated with respect to whether it can be added to the spreader set. Additionally, $\mathcal{F}(v_i)$ is equal to v_i, according to the definition of Formula (6).

In Algorithm 1, *Spreaders* is a list of seed nodes; *h_degree_sorted* is a list storing h-index and degree values, in which all nodes are arranged in descending order by h-index; and *degree_sorted* is a dictionary that stores information about all neighbor nodes of every node. The information in the dictionary is arranged in descending order by degree.

The first line of Algorithm 1 reads the network information. Line 2 to line 7 find the h-index of every node and store this information in a list. Line 8 sorts the list. Line 9 to line 11 select the appropriate nodes from the list as spreaders according to Formula (9). If there are two nodes with the same h-index, then the node with a higher degree is selected. If there are several nodes with the same h-index and the same degree simultaneously, then the one with higher coreness is selected. If these alternative nodes have the same h-index, degree and coreness, one is randomly selected.

Algorithm 2 reflects a special form of the h-index group centrality for trivial groups. The distance parameter *dis* is introduced to make the spreaders maintain the appropriate distance, which can effectively overcome the overlap problem.

Algorithm 1 H-Index Trivial Group Algorithm, HTGA

Input: network: $G(V, E)$; the number of spreaders: n.
Output: the spreaders
1: $degree\_sorted$ = loadData(filename)
2: **for** every member in $degree\_sorted$ **do**
3: $h = \max[h, i \cdot bool(d_i \geq i)]$
4: **if** h_degree_sorted did not append($this$) **then**
5: append
6: **end if**
7: **end for**
8: sort(h_degree_sorted)
9: **for** $i = 0$ to n **do**
10: $Spreaders$.append($h\_degree\_sorted[i]$)
11: **end for**
12: **return** *Spreaders*

Algorithm 2 H-Index Trivial Group with Distance Algorithm, HTGDA

Input: network: $G(V, E)$; distance: dis; the number of spreaders: n.
Output: the spreaders
1: $degree\_sorted$ = loadData(filename)
2: h_degree_sorted = CalcH($degree\_sorted$)
3: sort(h_degree_sorted)
4: **while** size of $Spreaders \neq n$ **do**
5: $u = \arg\max \{h_v, degree_v | v \notin Spreaders, v \notin Cover, v \in h\_degree\_sorted\}$
6: $Spreaders$.append(u)
7: **for** every v' in $\left\{v' \middle| u = u_{latest}, d_{u,v'} \leq dis, v' \in V\right\}$ **do**
8: $Cover$.append(v')
9: **end for**
10: **end while**
11: **return** *Spreaders*

Formula (10) defines μ as zero and defines group G_i as node v_i. It is a special form of Formula (6). Algorithm 2 is based on Formula (10). Additionally, $N^{(0)}(x) = x$, according to the definition of Formula (6).

$$\frac{\left|N^{(0)}(\mathcal{F}(v_i)) \cap N^{(dis)}(SP)\right|}{|N^{(dis)}(SP)|} \leq \mu, \quad \mu = 0. \tag{10}$$

The data structure is the same as in Algorithm 1. The new data structure $Cover$ is a dictionary for recording nodes that have been covered by dis. Nodes in $Cover$ cannot be selected into *Spreaders*. The first line reads the network information. Line 2 to line 3 calculate the h-index of every node and store the information in a sorted list. Line 4 to line 10 select spreaders. The update process of $Cover$ is shown in line 7 to line 8. Line 11 returns the selected *Spreaders* set.

Algorithm 3 reflects a special form of the h-index group centrality for trivial groups. In this algorithm, threshold $\mu \in (0, 1)$ here is for the purpose of reducing the repetition rate of neighbor nodes to ensure the propagation effect.

Formula (11) defines the range of μ as $(0, 1)$ and defines group G_i as node v_i. It is a special form of Formula (6). Algorithm 3 is based on Formula (11).

$$\frac{\left|N^{(1)}(\mathcal{F}(v_i)) \cap N^{(1)}(SP)\right|}{|N^{(1)}(SP)|} \leq \mu, \quad \mu \in (0, 1). \tag{11}$$

The specific steps of HTGDIA are as follows:

1) set a threshold μ, $\mu \in (0, 1)$;
2) keep updating *Spreaders* and neighbor node set *Neighbor*;
3) whenever a new node A attempts to join the *Spreaders* set, check Formula (11);
   a) if the new node A meets condition Formula (11), then A can join *Spreaders*; return to step 2);
   b) if not, then A cannot join *Spreaders*; investigate the other nodes;
4) if the size of *Spreaders* grows to n, then the algorithm terminates.

Algorithm 3 H-Index Trivial Group Deleting Intersection Algorithm, HTGDIA

Input: network: $G(V, E)$; the number of spreaders: n; intersection threshold: μ.
Output: the spreaders

$degree\_sorted$ = loadData(filename)
h_degree_sorted = CalcH($degree\_sorted$)
sort(h_degree_sorted)
while size of *Spreaders* $\neq n$ **do**
  $u = \arg\max \left\{ h_v, degree_v \middle| v \notin Spreaders, v \in h\_degree\_sorted, \frac{\left|N^{(1)}(\mathcal{F}(u)) \cap N^{(1)}(S)\right|}{\left|N^{(1)}(S)\right|} \leq \mu \right\}$
  Spreaders.append(u)
  for every v' in $\{v' | u = u_{latest}, d_{u,v'} = 1, v' \in V\}$ **do**
    $Neighbor$.append(v')
  end for
end while
return *Spreaders*

Algorithm 4 reflects a special form of the h-index group centrality for when a group is a community. We refer to such groups that are formed by community detection as community groups. This algorithm combines the h-index group centrality with community detection. The superiority of this algorithm on some networks was experimentally proven.

Formula (12) defines the range of μ as $[1, +\infty)$ and defines group G_i as community C_i. It is a special form of Formula (6). Algorithm 4 is based on Formula (12).

$$\begin{aligned} &\exists \mu, \mu \in [1, +\infty), \\ &\text{s.t. } \frac{\left|N^{(r)}(\mathcal{F}(C_i)) \cap N^{(n)}(SP)\right|}{|N^{(n)}(SP)|} \leq \mu. \end{aligned} \tag{12}$$

The specific steps of HCGA are as follows:

1) detect the communities using the Louvain algorithm;
2) sort communities in descending order according to the size of each community;
3) rank nodes in descending order of the h-index and degree within each community;
4) select the first node of the first to the nth communities to join the *Spreaders* set.

Algorithm 5 reflects a special form of the h-index group centrality for groups that are community groups. This algorithm combines the h-index group centrality with community detection and optimizes the method of selecting spreaders.

Algorithm 5 is based on Formula (12).

Algorithm 4 H-Index Community Group Algorithm, HCGA

Input: network: $G(V, E)$; the number of spreaders: n.
Output: the spreaders
1: *communities* = Louvain()
2: sort(*communities*) according to the size of every community
3: **for** every C_i in *communities* **do**
4: sort inner nodes according to h and degree
5: **end for**
6: **for** $i = 1$ to n **do**
7: *Spreaders*.append(v_{first} in C_i)
8: **end for**
9: **return** *Spreaders*

Algorithm 5 H-Index Community Group Summing Algorithm, HCGSA

Input: network: $G(V, E)$; the number of spreaders: n.
Output: the spreaders
1: *communities* = Louvain()
2: **for** every C_i in *communities* **do**
3: Calc $\sum_{p=1}^{q} h_p$, $q = |C_i|$
4: **end for**
5: sort(*communities*) according to $\sum_{p=1}^{q} h_p$
6: **for** every C_i in *communities* **do**
7: sort inner nodes according to h and degree
8: **end for**
9: **for** $j = 1$ to n **do**
10: *Spreaders*.append(v_{first} in C_j)
11: **end for**
12: **return** *Spreaders*

The specific steps of HCGSA are as follows:

1) detect the communities using the Louvain algorithm;
2) sort communities in descending order according to the sum of the h-index of all nodes in each community;
3) rank nodes in descending order of the h-index and degree within each community;
4) select the first node of the first to the nth communities to join the *Spreaders* set.

3.3. **Experimental steps.** To evaluate the h-index group centrality, we used two forms of group: node and community. The details of our algorithms are described in Section 3.2.

There are two fixed steps in the experiments: obtaining spreaders and spreading. If we build groups as communities, then additional community detection will be required before the two fixed steps. We describe the three experimental steps as follows.

1) The first step is community detection. We used the Louvain algorithm for community detection.
2) The second step is selecting spreaders. The innovations in our experiments were fully reflected in acquiring the spreader set. We performed experiments with this step using the five h-index group centrality algorithms and five comparison algorithms. The h-index group centrality algorithms performed better than the comparison algorithms for all experimental datasets.

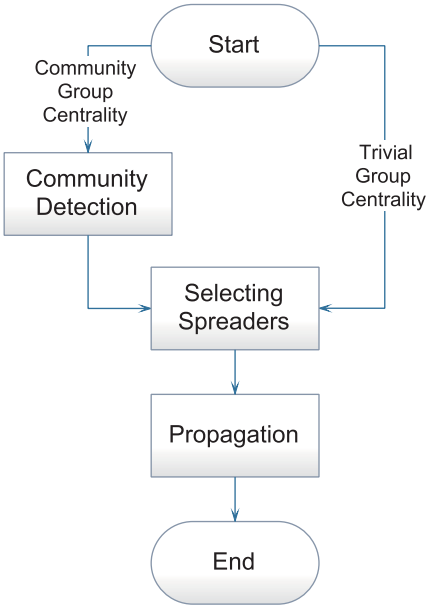


FIGURE 2. A flowchart of our methodology

3) The last step is propagation. We used a classic IC model in this step [35-37]. One of our comparison algorithms, the IO-ClusterRank algorithm, has assumed the susceptible infected recovered (SIR) model as its propagation model. However, another comparison algorithm, CCA, used the same IC model as our experiments. Additionally, some of the comparison algorithms, such as degree centrality and k-core, focused on centralities or structural characteristics of networks, but not on propagation, so there were no propagation models when they were published.

4. Evaluation and Discussion.

4.1. **Datasets.** We used four datasets to evaluate the performance of the h-index group centrality algorithms. First, we used the WikiTalk dataset, which contains all of the users and discussion from the inception of Wikipedia until January 2008. Second, we used the Email dataset, which contains all email data from a large European research institution over 18 months. We then used the Amazon dataset, which was collected by crawling the Amazon website. Finally, we used the DBLP dataset, which contains information regarding whether two computer scientists published at least one paper together. These four experimental datasets could be found at the Stanford Large Network Dataset Collection [38].

It can be seen from Table 1 that the attributes of different nodes in the Amazon network dataset have few differences. This feature of the Amazon dataset will play a significant role in Section 4.3. Additionally, the DBLP network was the dataset used when the CCA was published. The detailed information concerning these datasets is shown in Table 1 and Table 2.

4.2. **Comparison algorithms.** Our paper investigates a new centrality, the h-index group centrality. We compared this centrality with some heuristic algorithms and experimentally proved its superior effect in identifying influential nodes. There were two centrality algorithms and three other heuristic algorithms that served as comparison algorithms in our experiments. The details are shown in Table 3.

The advantage of the degree centrality algorithm is that it has a wide range of applications and its disadvantage is not novel enough. Compared to degree centrality algorithm,

Table 1. Basic information of datasets

	WikiTalk	*Email*	*Amazon*	*DBLP*
Nodes	2394385	265214	410236	317080
Edges	5021410	420045	3356824	1049866
Average degree	2.10	1.58	8.18	3.31
Max degree	100022	930	10	306
Average coreness	1.84	2.00	6.04	2.96
Max coreness	100012	730	10	269
Average h-index	0.36	1.06	3.46	1.28
Max h-index	868	153	9	59

In this table, degree means out degree, and our h-index group centrality is based on out degree.

Table 2. Simple analysis of datasets

	WikiTalk	*Email*	*Amazon*	*DBLP*
α_1	2.1×10^{-5}	1.7×10^{-3}	8.1×10^{-1}	1.1×10^{-2}
α_2	1.8×10^{-5}	2.7×10^{-3}	6.0×10^{-1}	1.1×10^{-2}
α_3	4.1×10^{-4}	6.9×10^{-3}	3.8×10^{-1}	2.1×10^{-2}
α_4	0.0526	0.0671	0.4064	0.6324
α_5	9	14	20	21

α_1 refers to $\frac{Average\ degree}{Max\ degree}$; α_2 refers to $\frac{Average\ coreness}{Max\ coreness}$; α_3 refers to $\frac{Average\ h\text{-}index}{Max\ h\text{-}index}$; α_4 refers to the average clustering coefficient; and α_5 refers to the diameter (longest shortest path).

Table 3. Comparison algorithms

Algorithm	Description
Degree centrality	Published algorithm. Select spreaders by degree.
Degree group centrality	An algorithm based on the extended definition in Section 3.1.1. Select spreaders by degree but consider the concept of a group.
k-core	Published algorithm. Select spreaders by coreness.
Core covering algorithm (CCA) [13]	Published algorithm. Select spreaders by coreness but consider the distance.
IO-ClusterRank [20]	Published algorithm. Select spreaders by IO-ClusterRank.

degree group centrality algorithm is a more novel algorithm combining group with degree. Degree group centrality algorithm is not a published algorithm so that it can only be used as an auxiliary comparison algorithm in experiments. The k-core algorithm and the core covering algorithm are both core-based algorithms. Compared with degree, coreness can reveal the structural and hierarchical characteristics of networks. Therefore, the core-based algorithms can be applied to some networks where the degree-based algorithms do

not perform well. The core-based algorithms have the disadvantage of higher computational complexity compared with the degree-based algorithms so that the core-based algorithms are not suitable for dynamic large-scale networks. As the most novel algorithm in comparison algorithms, IO-ClusterRank algorithm retains lower computational complexity based on original centrality but it cannot achieve excellent results in networks with uniform out/in-degree distribution [20].

4.3. **Results.** We use the final number of infected nodes in a network to evaluate the effect of an algorithm. Figures 3, 4, 5 and 6 display the final results of our experimental algorithms and comparison algorithms. The percentage in each graph represents the percentage of nodes that are eventually infected on the network to the total network nodes. The X coordinate axis in every subfigure of Figures 3, 4, 5 and 6 refers relative to the number of spreaders, and the Y coordinate axis refers to the number of final infected nodes. The rectangular shaded boxes in Figures 3, 4, 5 and 6 are enlargements of parts of polylines.

Parameter p which is between 0 to 1 represents the propagation probability of the IC model. We learnt that our comparison algorithm CCA performs better [13] with a large p than with a small p. In order to give full play to the characteristics of each algorithm under fair conditions, we decided to adopt a series of large p-values in our experiments. Specific experimental parameters setting is indicated in Figure 3, Figure 4, Figure 5 and Figure 6. Parameter dis is set as 2 in our experiments. The parameter dis mainly affects CCA and HTGDA. CCA is one of the comparison algorithms. According to the CCA paper [13], we know that CCA performs better when dis is 2 than when dis is 1. We chose to set dis to 2 to make the comparison algorithm CCA play a better role for a more convincing contrast effect.

The average time complexity of sorting is $O(n \log n)$ because we used quick sort. The time complexity of selecting nodes of HTGA is $O(n)$. Therefore, the time complexity of HTGA is $O(n \log n)$. The optimal time complexity of HTGA can reach $O(n)$ if we use non-comparison sort algorithms. For the same reason, the optimal time complexities of HTGDA and HTGDIA can reach $O(n)$. Furthermore, the time complexities of HCGA and HCGSA are consistent with the time complexity of the Louvain community discovery algorithm.

The corresponding experiments for every subfigure in Figures 3, 4, 5 and 6 are as follows.

Figure 3(a) to Figure 3(f) are experimental results based on the WikiTalk dataset. HTGA is the best of the five h-index group centrality algorithms in this case. Regarding the reasons for this, the WikiTalk network exhibits the long tail characteristics we discussed in Section 3.1. The best results of HTGA exceed the best results of the state-of-the-art comparison algorithm by 11.0%, 18.0%, and 41.8% under different p value of IC model. Figure 3(a) to Figure 3(c) are comparisons between HTGA and the five comparison algorithms, and Figure 3(d) to Figure 3(f) are comparisons among the five experimental h-index group centrality algorithms.

Figure 4(a) to Figure 4(f) are experimental results based on the Email dataset. HTGDA is the best of the five h-index group centrality algorithms in this case. The reason for this is that the Email network includes a tail where some of the y-coordinates of its nodes are between 1 and 10, which we discussed in Section 3.1. For example, if the ordinate value of one node at the tail is 10, then this indicates that there are 10 nodes with the same h-index: in other words, we need an effective algorithm to distinguish which one of the 10 nodes is the most influential. We investigate whether there are some successor nodes of existing spreaders among them. If there are such nodes, then we think that they have a large

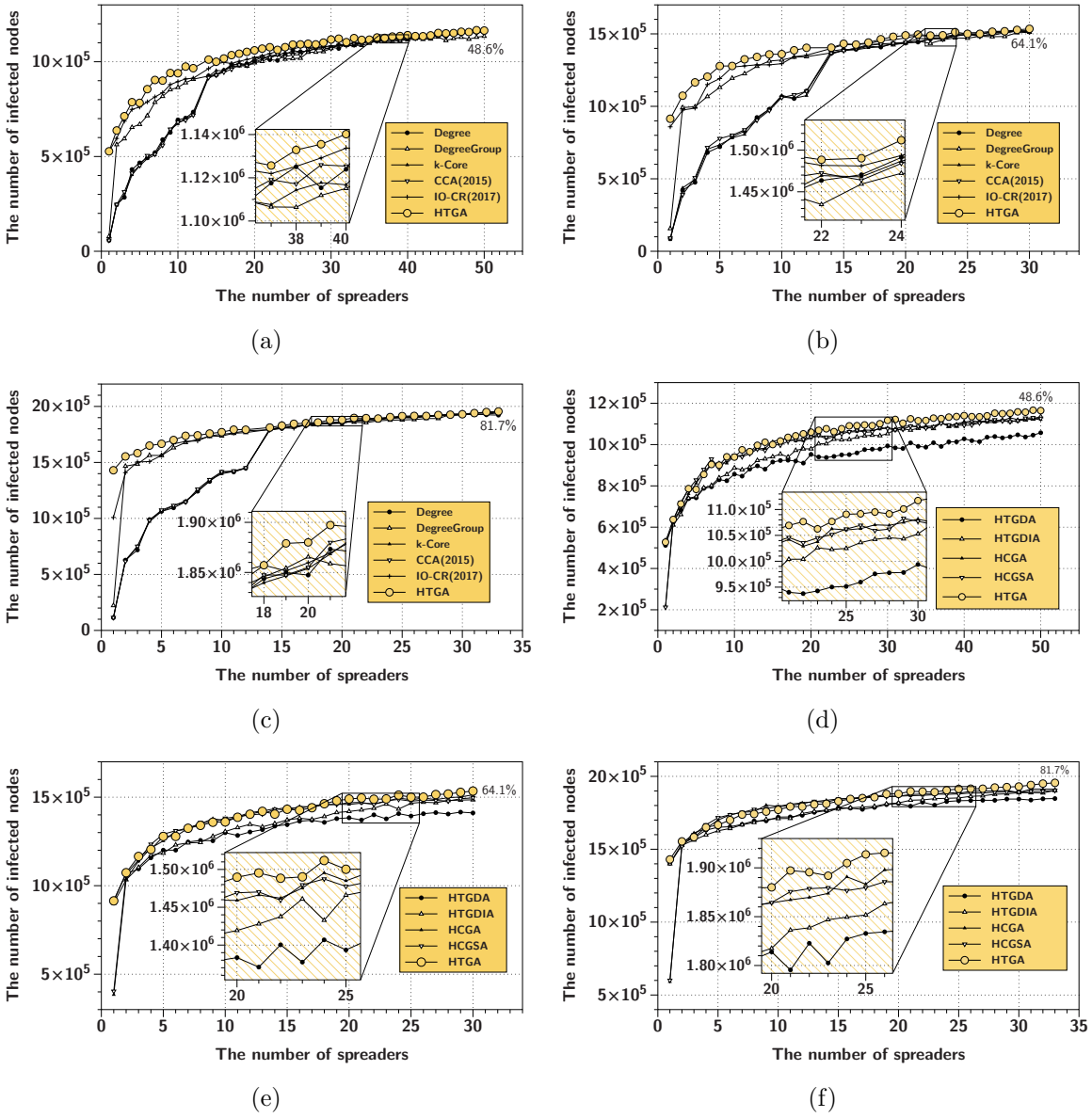


FIGURE 3. Experimental results of the WikiTalk dataset. The best algorithm in this dataset is HTGA, so we focus on HTGA. According to Section 3.1 and Section 3.2, the parameter μ in HTGA could always be set to 1. Parameters (without μ) setting: (a) $p = 0.5$, $dis = 2$; (b) $p = 0.7$, $dis = 2$; (c) $p = 0.9$, $dis = 2$; (d) $p = 0.5$, $dis = 2$; (e) $p = 0.7$, $dis = 2$; (f) $p = 0.9$, $dis = 2$.

overlapping area with the propagation range of existing spreaders. We believe that their influence is reduced, and they cannot be selected. HTGDA reflects this idea. The best results of HTGDA exceed the best results of the state-of-the-art comparison algorithm by 21.4%, 20.0%, and 22.8%. Figure 4(a) to Figure 4(c) are comparisons between HTGDA and the five comparison algorithms, and Figure 4(d) to Figure 4(f) are comparisons among the five experimental h-index group centrality algorithms.

Figure 5(a) to Figure 5(f) are experimental results based on the DBLP dataset. HTGDIA is the best of five h-index group centrality algorithms in this case. The tail of the DBLP network and the tail of the Email network have similar characteristics. In both

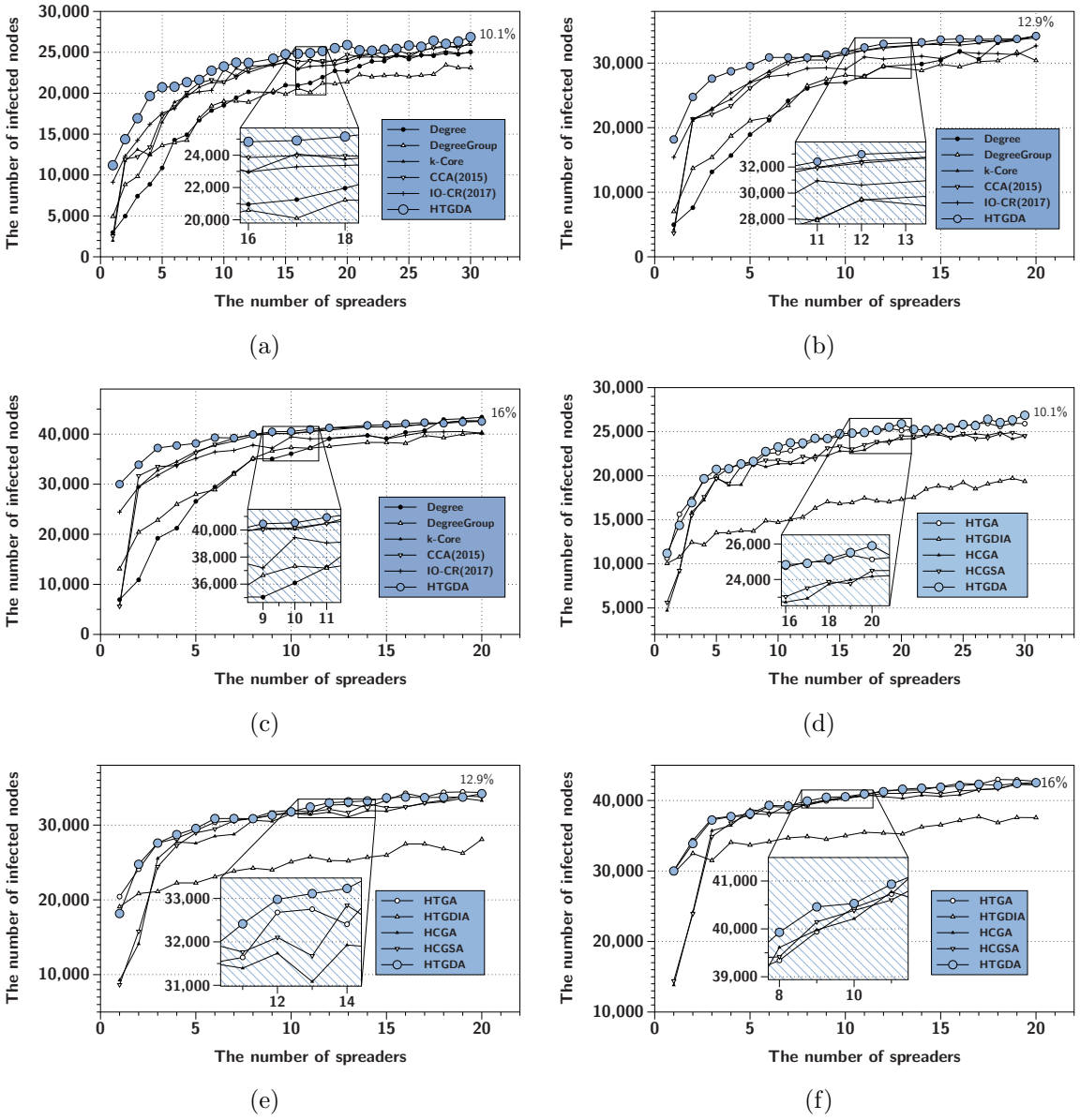


FIGURE 4. Experimental results of the Email dataset. The best algorithm in this dataset is HTGDA, so we focus on HTGDA. According to Section 3.1 and Section 3.2, the parameter μ in HTGDA could always be set to 0. Parameters (without μ) setting: (a) $p = 0.5$, $dis = 2$; (b) $p = 0.7$, $dis = 2$; (c) $p = 0.9$, $dis = 2$; (d) $p = 0.5$, $dis = 2$; (e) $p = 0.7$, $dis = 2$; (f) $p = 0.9$, $dis = 2$.

networks, it is necessary to determine which node has the most influence among several nodes with the same h-index. However, the two networks are still different. For example, the h-index value of the DBLP network is generally lower than the h-index value of the Email network, and the α_2 of DBLP and the α_2 of Email are not on the same order of magnitude in Table 2. The fit of HTGDIA to the Email network is less suitable than that for the DBLP network. Considering that these two networks have some of the same characteristics, we adopted the same ideas for these two networks but made some modifications to the specific algorithm details on DBLP, and then, we proposed HTGDIA, which fits DBLP. The best results of HTGDIA exceed the best results of the

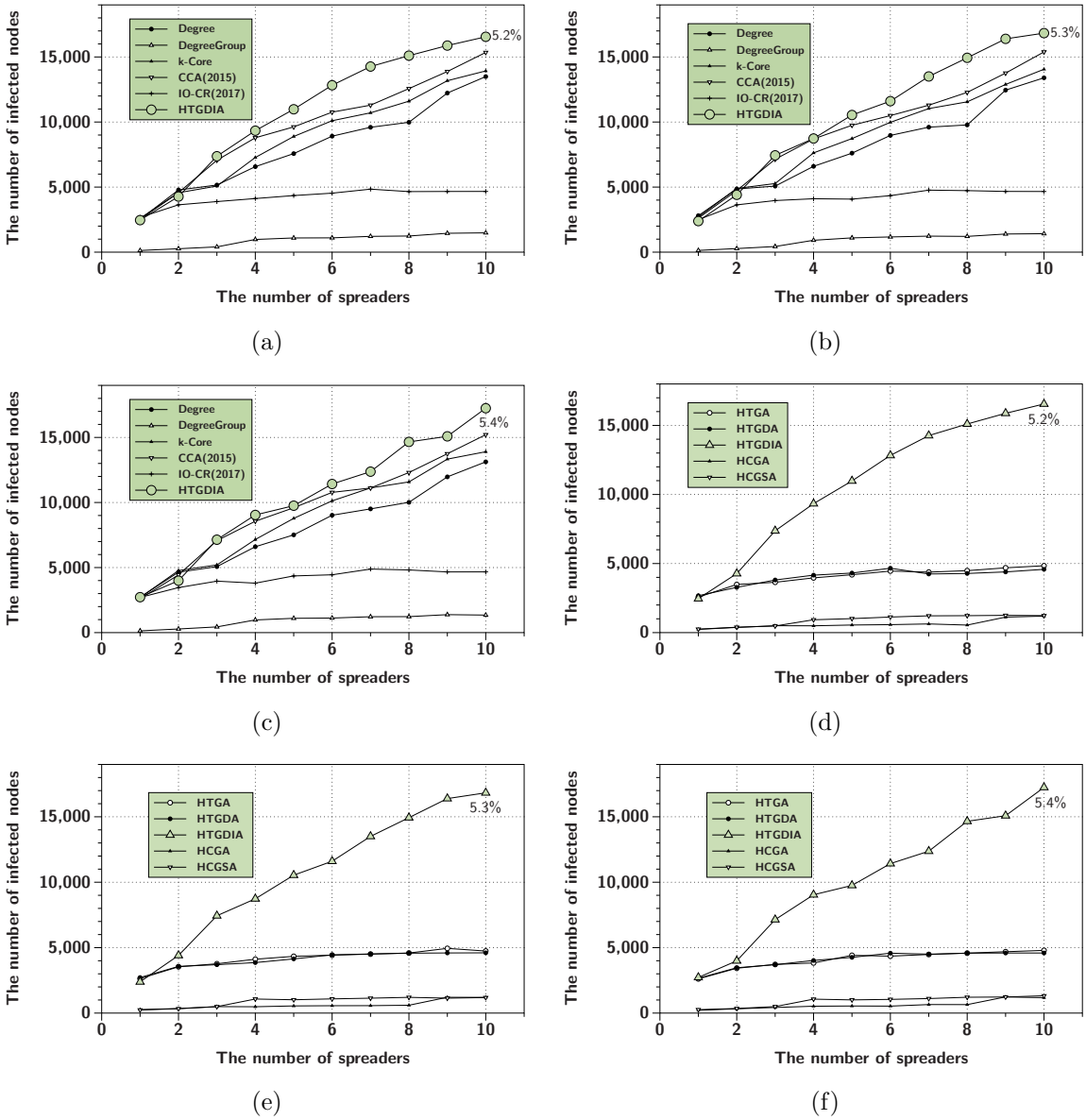
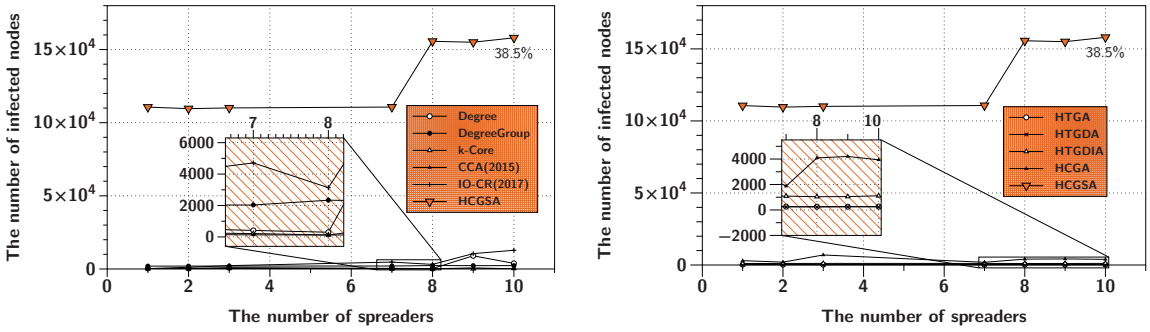


FIGURE 5. Experimental results of the DBLP dataset. The best algorithm in this dataset is HTGDIA. Parameters setting: (a) $p = 0.9$, $dis = 2$, $\mu = 0.03$; (b) $p = 0.9$, $dis = 2$, $\mu = 0.04$; (c) $p = 0.9$, $dis = 2$, $\mu = 0.05$; (d) $p = 0.9$, $dis = 2$, $\mu = 0.03$; (e) $p = 0.9$, $dis = 2$, $\mu = 0.04$; (f) $p = 0.9$, $dis = 2$, $\mu = 0.05$.

state-of-the-art comparison algorithm by 26.3%, 21.6%, and 19.1%. Figure 5(a) to Figure 5(c) are comparisons between HTGDIA and the five comparison algorithms, and Figure 5(d) to Figure 5(f) are comparisons among the five experimental h-index group centrality algorithms.

Figure 6(a) and Figure 6(b) are experimental results based on the Amazon dataset. HCGSA is the best of the five h-index group centrality algorithms in this case. In the tail of the Amazon network, there are nodes with ordinate values exceeding 74,000. In other words, we need to select the most influential node among tens of thousands of nodes with the same h-index value instead of choosing among ten nodes like the Email dataset and the DBLP dataset. Our idea for networks like the Amazon dataset is to divide and



(a) (b)

FIGURE 6. Experimental results of the Amazon dataset. The best algorithm in this dataset is HCGSA, so we focus on HCGSA. According to Section 3.1 and Section 3.2, the parameter μ in HCGSA could always be set to 1. Parameters (without μ) setting: (a) $p = 0.9$, $dis = 2$; (b) $p = 0.9$, $dis = 2$.

conquer. We divided the entire network into communities through community detection, narrowed the scope of the problem, and proposed algorithms like HCGA and HCGSA. Experiments have proven that our idea works. The best result of HCGSA exceeds the best result of the state-of-the-art comparison algorithm by 1205.3 times. Figure 6(a) shows a comparison between HCGSA and the five comparison algorithms, and Figure 6(b) shows a comparison among the five experimental h-index group centrality algorithms.

The h-index group centrality method is scalable in terms of adapting to different network scales. Meanwhile, the method combining the h-index group centrality with community detection has enormous potential.

4.4. **Discussion.** There are many future study directions based on the h-index group centrality. First, the current value range of the h-index is $h \in \mathbb{Z}$, $h > 0$, which could be extended to $h' \in \mathbb{R}$, $h' > 0$. This extension can help us better distinguish the most influential nodes. Second, the h-index group centrality can be expanded into the h-index hierarchical centrality. In future research, we will explore whether it is necessary to aggregate groups into zones and zones into clusters. Third, we will explore the relationship among the h-index, degree, and coreness. [18] conducted by Lü et al. shows that the h-index is related to the coreness and degree. Defining an operator and using it repeatedly can change the results from degree to h-index to coreness. We can explore the relationship among them based on this research and analyze the impact of this connection on propagation. This connection can extend the topological properties of the h-index group centrality, which may help locate influential nodes in dynamical networks [39].

5. **Conclusion.** H-index group centrality algorithms were experimentally proven to be scalable for identifying influential nodes. The scalability of our method was reflected in two respects. First, our algorithms met the needs of large-scale network datasets including from 200,000 nodes to 2,000,000 nodes. Second, our method performed well in terms of accuracy for spreader sets of different sizes from 1 to 50. Even in bad cases, the results of our method exceeded the best results of compared methods by 2.6%. Furthermore, by investigating the h-index group centrality, we can draw the conclusion that the combination of the h-index group centrality and community detection produces an exciting effect and offers great potentiality.

The h-index group centrality method contributes to not only theoretical research on influence maximization but also practical applications such as conventional routing algorithms, network security, selection of convergence points for multi-source multicasts, mobile edge intelligence-enabled networking [40], preventive medicine, and many other applied fields worth anticipating.

Acknowledgment. This work is partially supported by Jun Zhao, Chuan Zhou and Mengchi Xing at the Chinese Academy of Sciences. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] X. Zhao, H. Zhang, M. Zhang, L. Cheng and S. Ma, Identifying influential nodes in large-scale software networks, *ITOEC*, Chongqing, pp.764-767, 2017.
[2] Q. Liu, Z. Ma, X. Yang and H. Zhu, Identifying influential actors in social network platforms, *CIS*, Shenzhen, pp.223-226, 2015.
[3] H. A. Khanday, A. H. Ganai and D. R. Hashmy, A comparative analysis of identifying influential users in online social networks, *ICSNS*, Coimbatore, pp.1-6, 2018.
[4] N. Hafiene and W. Karoui, A new structural and semantic approach for identifying influential nodes in social networks, *AICCSA*, Hammamet, pp.1338-1345, 2017.
[5] Y. Kim and N.-W. Cho, Research trends in social network analysis using topic modeling and network analysis, *ICIC Express Letters*, vol.12, no.1, pp.71-78, 2018.
[6] T. Iwata, A. Shah and Z. Ghahramani, Discovering latent influence in online social activities via shared cascade poisson processes, *ACM SIGKDD*, Chicago, IL, USA, pp.266-274, 2013.
[7] W. Kang, G. Tang, Y. Sun and S. Wang, Identifying influential nodes in complex network based on weighted semi-local centrality, *ICCC*, Chengdu, pp.2467-2471, 2016.
[8] M. Richardson and P. Domingos, Mining knowledge-sharing sites for viral marketing, *ACM SIGKDD*, Edmonton, Alberta, Canada, pp.61-70, 2002.
[9] D. Kempe, J. Kleinberg and É. Tardos, Maximizing the spread of influence through a social network, *ACM SIGKDD*, Washington, D.C., pp.137-146, 2003.
[10] L. C. Freeman, Centrality in social networks' conceptual clarification, *Social Networks*, vol.1, no.3, pp.215-239, DOI: 10.1016/0378-8733(78)90021-7, 1978.
[11] T. Agryzkov et al., An algorithm for ranking the nodes of an urban network based on the concept of PageRank vector, *Applied Mathematics and Computation*, pp.2186-2193, DOI: 10.1016/j.amc.2012.08.064, 2012.
[12] T. Agryzkov et al., A new betweenness centrality measure based on an algorithm for ranking the nodes of a network, *Applied Mathematics and Computation*, pp.467-478, DOI: 10.1016/j.amc.2014.07.026, 2014.
[13] J. Cao et al., A k-core based algorithm for influence maximization in social networks, *Chinese Journal of Computers*, vol.38, no.2, pp.238-248, DOI: 10.3724/SP.J.1016.2015.00238, 2015.
[14] J. Niu, J. Fan, L. Wang and M. Stojinenovic, K-hop centrality metric for identifying influential spreaders in dynamic large-scale social networks, *IEEE Global Communications Conference*, Austin, TX, pp.2954-2959, 2014.
[15] Z. Liang and J. Li, Identifying and ranking influential spreaders in complex networks, *ICCWAMTIP*, Chengdu, pp.393-396, 2014.
[16] W. Chen, Y. Wang and S. Yang, Efficient influence maximization in social networks, *ACM SIGKDD*, Paris, France, pp.199-208, 2009.
[17] J. Leskovec, A. Krause, C. Guestrin et al., Cost-effective outbreak detection in networks, *ACM SIGKDD*, San Jose, CA, USA, pp.420-429, 2007.
[18] L. Lü, T. Zhou, Q.-M. Zhang and H. E. Stanley, The H-index of a network node and its relation to degree and coreness, *Nature Communications*, vol.7, p.10168, DOI: 10.1038/ncomms10168, 2016.
[19] J. Goldenberg and L. E. Muller, Talk of the network: A complex systems look at the underlying process of word-of-mouth, *Marketing Letters*, vol.12, no.3, pp.211-223, DOI: 10.2307/40216600, 2001.
[20] Y. Wang, G. Yan, Q. Ma, Y. Wu and D. Jin, Identifying influential spreaders on weighted networks based on ClusterRank, *ISCID*, Hangzhou, pp.476-479, 2017.

[21] S. B. Seidman, Network structure and minimum degree, *Social Networks*, vol.5, no.3, pp.269-287, DOI: 10.1016/0378-8733(83)90028-X, 1983.
[22] K. Shin, T. Eliassi-Rad and C. Faloutsos, CoreScope: Graph mining using k-core analysis – Patterns, anomalies and algorithms, *ICDM*, Barcelona, pp.469-478, 2016.
[23] Y. Liu, M. Tang, J. Yue and J. Gong, Identify influential spreaders in complex real-world networks, *UIC-ATC-ScalCom*, Beijing, pp.1144-1148, 2015.
[24] J. E. Hirsch, An index to quantify an individual's scientific research output, *Proc. of the National Academy of Sciences*, vol.102, no.46, pp.16569-16572, DOI: 10.1073/pnas.0507655102, 2005.
[25] P. Wijegunawardana, K. Mehrotra and C. Mohan, Finding rising stars in heterogeneous social networks, *ICTAI*, San Jose, CA, pp.614-618, 2016.
[26] R. Hanafy, S. Makady and A. EIKorany, A social trust metric for scholarly reputation mining, *ASONAM*, Barcelona, pp.61-68, 2018.
[27] V. P. M. Freire and D. R. Figueiredo, Ranking in collaboration networks using a relationship intensity metric, *Brazilian Symposium on Collaborative Systems – Simposio Brasileiro de Sistemas Colaborativos*, Belo Horizonte, pp.71-78, 2010.
[28] Y. Wang, G. Yan, Q. Ma, Y. Wu and M. Zhang, Identifying influential nodes based on vital communities, *DASC/PiCom/DataCom/CyberSciTech*, Athens, pp.314-317, 2018.
[29] M. Lei and D. Wei, Identifying influence for community in complex networks, *CCDC*, Shenyang, pp.5346-5349, 2018.
[30] V. D. Blondel et al., Fast unfolding of communities in large networks, *Journal of Statistical Mechanics: Theory and Experiment*, vol.2008, no.10, pp.155-168, DOI: 10.1088/1742-5468/2008/10/p10008, 2008.
[31] S. Ghosh et al., Distributed Louvain algorithm for graph community detection, *IPDPS*, Vancouver, BC, pp.885-895, 2018.
[32] G. Lin et al., Community detection in power grids based on Louvain heuristic algorithm, *EI2*, Beijing, pp.1-4, 2017.
[33] K. Saito et al., Efficient discovery of influential nodes for SIS models in social networks, *Knowledge and Information Systems*, vol.30, no.3, pp.613-635, DOI: 10.1007/s10115-011-0396-2, 2012.
[34] L. Zhong and F. Lv, An improved PageRank for identifying the influential nodes based on resource allocation in directed networks, *ICCWAMTIP*, Chengdu, pp.42-45, 2017.
[35] A. Kumari and S. N. Singh, Online influence maximization using rapid continuous time independent cascade model, *International Conference on Cloud Computing, Data Science & Engineering – Confluence*, Noida, pp.356-361, 2017.
[36] X. Yu and T. Chu, Learning the structure of influence diffusion in the independent cascade model, *CCC*, Dalian, pp.5647-5651, 2017.
[37] W. Yang, L. Brenner and A. Giua, Influence maximization in independent cascade networks based on activation probability computation, *IEEE Access*, vol.7, pp.13745-13757, DOI: 10.1109/ACCESS.2019.2894073, 2019.
[38] *Stanford Large Network Dataset Collection*, http://snap.stanford.edu/data/, Accessed in April 2019.
[39] J. Wang, X. Feng, Z. Yin, B. Guo and Z. Zheng, Identifying influential nodes in Boolean networks through dynamical voter rank, *ITNEC*, Chengdu, pp.1016-1020, 2017.
[40] Y. Sheng, The challenges and opportunities of mobile edge intelligence – Enabled networking, *Journal of Network New Media*, vol.7, no.4, pp.1-4, DOI: CNKI:SUN:WJSY.0.2018-04-001, 2018.