information

MDPI

Article

Identifying Influential Nodes in Complex Networks Based on Local Effective Distance

Junkai Zhang, Bin Wang, Jinfang Sheng *, Jinying Dai, Jie Hu and Long Chen

School of Computer Science and Engineering, Central South University, Changsha 410083, China; 174611108@csu.edu.cn (J.Z.); wb_csut@csu.edu.cn (B.W.); 174712058@csu.edu.cn (J.D.); hujie1995@csu.edu.cn (J.H.); lgchen@csu.edu.cn (L.C.)

* Correspondence: jfsheng@csu.edu.cn

Received: 11 September 2019; Accepted: 8 October 2019; Published: 10 October 2019

Abstract: With the rapid development of Internet technology, the social network has gradually become an indispensable platform for users to release information, obtain information, and share information. Users are not only receivers of information, but also publishers and disseminators of information. How to select a certain number of users to use their influence to achieve the maximum dissemination of information has become a hot topic at home and abroad. Rapid and accurate identification of influential nodes in the network is of great practical significance, such as the rapid dissemination, suppression of social network information, and the smooth operation of the network. Therefore, from the perspective of improving computational accuracy and efficiency, we propose an influential node identification method based on effective distance, named KDEC. By quantifying the effective distance between nodes and combining the position of the node in the network and its local structure, the influence of the node in the network is obtained, which is used as an indicator to evaluate the influence of the node. Through experimental analysis of a lot of real-world networks, the results show that the method can quickly and accurately identify the influential nodes in the network, and is better than some classical algorithms and some recently proposed algorithms.

Keywords: Influential nodes; complex networks; effective distance; total influence

1. Introduction

The Graph represents various real-world networks through points and lines, which has become a universal method to study complex networks. An abstract graph shows that the real-world network has no network properties such as node size, position, and function [1]. Any system in the real or virtual world can form an abstract graph G(V, E) through nodes, set V, and node connected edges, set E. Much social interaction phenomena in the real world can be represented as complex networks [2,3]. For instance, if we take instant messaging tools (QQ, WeChat, etc.), social networking sites (Facebook, Weibo, etc.), and community BBS (Douban, Tianya, etc.), nodes in the network represent users and edges represent social relations between users. An individual with strong social influence can cause or urge another user to produce the corresponding behavior through his or her behavior [4]. While transmitting some good values, there is also a production of a lot of negative effects, such as the rapid spread of infectious diseases [5,6], computer virus transmission [7,8], urban-spread rumors [9,10], power network collapse [11], and so on. Once the important node is attacked or loses efficacy, it will cause immeasurable loss to the normal operation of the network [12]. For example, in 1965, a small 4-square-inch relay failure in northeast North America directly led to a large-scale power supply failure event. In 2009, ethnic separatists incited incidents of beating, smashing, looting, and arson through the Internet and other channels. In 2012, rumors about the end of the world led to panic buying of candles. In 2017, computational ransomware viruses swept

through some banks, schools, medical institutions, and other institutions in dozens of countries around the world. It is possible to reduce or even avoid economic losses and disasters if influential nodes in these networks can be identified in advance and preventive control can be carried out. Therefore, it is particularly important to identify influential nodes in the network and take relevant measures. The existing classical ranking methods all have their advantages and disadvantages, which are mainly divided into the following three situations: (i) The accuracy of node ranking is not enough. Some algorithms only consider one attribute of the node, which may be misarranged; (ii) High computational time complexity. Most algorithms based on path and node removal involve the calculation of the shortest path. Although the sorting is relatively accurate, the time complexity is too high to be suitable for large-scale networks. In addition, some algorithms also restrict the nature of the network; (iii) Not consistent with the actual situation. In an undirected and unweighted network, the traditional node detection method treats the distance between adjacent nodes as equal to 1. However, the connecting edges between different nodes play different roles in the process of information transmission, which makes the evaluation result different from the actual situation [13].

Aiming at the above problems, this paper proposes an influential node identification method based on effective distance, which uses the node's attributes and interaction with neighboring nodes to comprehensively evaluate the importance of nodes in the network [14,15]. This method not only solves the problem that the importance of the node is not accurate enough by local attribute evaluation, but also solves the problem that the operation scale is too large considering the excessive path, especially when calculating the distance between nodes—the "effective distance" is introduced to make the identification the algorithm is more realistic. In general, the algorithm has certain advantages over other classical algorithms.

The main contributions of this work are as follows: (1) Accurate influential node detection: The KDEC method comprehensively considers a variety of attributes to sort the nodes, which can detect the influential nodes more effectively and obtain more accurate sorting results; (2) Parameter free: KDEC does not rely on prior knowledge and parameter adjustments but can automatically identify influential nodes; (3) Scalability: KDEC uses two-stage neighbors of nodes to identify influential nodes, which greatly reduces the calculation cost, and has no strict requirements on the global topology of the network. Therefore, it is suitable for connected or disconnected networks.

The rest of this article is organized as follows. Section 2 introduces the classical methods of influential node detection in complex networks and introduces the advantages and disadvantages of various algorithms according to classification. Section 3 introduces the design principle of KDEC ranking algorithm proposed in this paper and introduces the calculation process of the KDEC model in combination with a simple network. Section 4 verifies the effectiveness of the proposed method by experimental comparison with the classical algorithm and the recently proposed profitleader algorithm. Section 5 is the conclusion.

2. Related Work

In recent years, research on influential nodes in complex networks has always been the focus of current network research [16–18]. Many approaches based on a graph [19,20] have been established by scholars. These methods measure the influential nodes from different viewpoints and have a good performance in their respective application fields, but they also have their shortcomings and limitations. This paper briefly introduces some common methods based on structural centralities and iterative refinement centralities. For details, please refer to reference [21].

Structural centralities. Structure is one of the main characteristics of a network, through which we can find the connections between nodes to determine their influence in the network. Existing structure-based approaches can be divided into two categories: One type is based on neighborhood of nodes, such as Degree centrality [22], K-shell decomposition [23], H-index [24], etc.; the other is based on path between nodes, such as Eccentricity [25], Closeness centrality [26], Katz centrality [27], Betweenness centrality [28], Flow betweenness centrality [29], etc. Degree centrality

directly considers the number of neighbor nodes, and the computation time complexity is very low, but it ignores the topological connection relationship between nodes. K-shell is a degree-based coarse-grained ranking method, which can be regarded as a degree centrality optimization. After the node with a small degree of removal is removed, the remaining nodes are located at the center of the network. These nodes are more important than others. However, the influence of nodes at the same level is difficult to distinguish [30]. The H-index method is a compromise method that uses node strength or the degree to measure centrality. However, this method has a huge drawback, that is, the value of the edge weight needs to be in the appropriate range, otherwise the effective ranking result cannot be obtained. Eccentricity is generally believed that the shorter the distance between a node and all other nodes, the greater the centrality. The disadvantage is that it is susceptible to special values. Betweenness centrality and Flow betweenness centrality consider the two extremes, the former only considers the shortest path; while the latter considers all paths, but the time complexity is high. Unlike closeness centrality, that considers only the shortest path lengths between the node pairs, Katz centrality computes the influences of nodes by considering all the paths in the network, so it is not suitable for large-scale and disconnected networks.

Iterative refinement centralities. The iterative-based methods mainly include Eigenvector centrality [31], Cumulative nomination [32], PageRank [33], LeaderRank [34], HITS [35], and so on. These methods not only consider the number of neighbor nodes but also consider the influence of each neighbor node itself. The eigenvector centrality assumes that the importance of a node depends on both the number of neighbor nodes and their importance, and iteratively calculates the influence of each node. The disadvantage of this method is that it is not suitable for networks with tight connections. The cumulative nomination scheme assumes that more central individuals are nominated more frequently in the social network and considers the nomination values of each node and its neighbors. The iterative process needs to add an ever-changing parameter value. PageRank is the core algorithm of the Google search engine, which needs to introduce a random jump probability. The convergence speed of the model will be accelerated with the increase of jump probability, and the accuracy will also be reduced, thus leading to inconsistent ranking results. LeaderRank outperforms PageRank, but it is limited to performing well in directed networks and cannot be applied to undirected networks. The HITS method gives each node two metrics, which represent the influence of the node itself and its ability to propagate information in the network. The two metrics eventually converge through iteration. However, HITS is not suitable for networks with a wide range of tightly connected communities, because the close relationship between the nodes in the community will enhance the authoritative values and hub values of these nodes, and so topic draft [36].

3. The KDEC Method

At present, many scholars have proposed various influential node detection methods from different perspectives, each with their advantages and disadvantages. The influence of a node in a complex network mainly depends on three aspects. The first is the location of the node in the network. While the node is in the center of the network, the impact will be significant. Else, if the node is at the edge of the network, the impact will certainly be relatively small. The second is the number of neighbors around the node. The more nodes that can be affected in the local scope, the greater the influence of the nodes. The third is the ability of nodes to influence their neighbors. The smaller the distance between the node and the neighboring node, the greater the probability that the node transmits information to the neighboring node, and the greater the probability that the neighboring node is affected. On this basis, this paper proposes the KDEC method. As described in Section 1, the KDEC algorithm has the advantages of accurate impact-node detection, parameter freedom, and scalability. Compared with most existing algorithms, our algorithm performs well on various networks.

3.1. Preliminaries

Definition 1 (Degree). *The degree of a node refers to the sum of its edges, which is one of the most basic attributes in complex networks. The basic idea is that the higher the node degree is, the more neighbor nodes there are, and the larger its influence in the network. Given an undirected and unweighted network G, set $A = \{A_{ij}|i, j = 1...N\}$ represents the adjacency matrix of G, the degree of a node is defined as follows:*

$$d(v_i) = \sum_{j=1}^{N} a_{ij}, \tag{1}$$

where $d(v_i)$ denotes the degree of the node v_i and a_{ij} represents the element of A.

Definition 2 (K_core). *Given $G = (V, E)$, a k_core is a maximal subgraph that contains nodes of degree k or more, the core number of a node is the largest value k of a k_core containing that node. The core number is defined as follows:*

$$K\_core = core\_number(G). \tag{2}$$

Notice, for directed graphs, the node degree is defined to be the sum of in-degree and out-degree, and the k_core is not implemented for graphs with self-loops or parallel edges.

Definition 3 (Weight). *K-shell and Degree are two attributes of node v_i, and their synthetic is used as an indicator to measure the initial weight of the node v_i itself, which is expressed as Weight(v_i).*

$$Weight(v_i) = Ks(v_i) \times d(v_i), \tag{3}$$

where $Ks(v_i)$ represents the kernel of the node v_i and $d(v_i)$ represents the degree centrality of the node v_i. There are many indexes to measure the attributes of nodes, but a single attribute may not accurately describe the influence of nodes in the network, so we express it by combining attributes. The importance of a node mainly depends on its position in the network and the number of neighbors it can influence. Therefore, we choose to combine k_core and DC to represent the weight of a node.

Definition 4 (Effective Distance). *Given $G = (V, E)$, some methods automatically set the distance between two adjacent nodes to 1, this is the most basic and simplest way to calculate the distance between nodes, but it works against the principle in a real-world network where information flows interact. After the information passes through a node, the information flow will be assigned to different paths according to the importance of the inherent attributes of the node. In KDEC algorithm, we mainly consider node degree as the attribute of measuring information distribution. Since the probability of receiving information in the process of transmission varies with the node degree, the probability of different neighbor nodes being affected is different. The lower the degree value, the greater the probability that the node is affected by the neighbor node, which reflects that the closer the relationship between the node and the neighbor node is, the shorter the effective distance [13,37] between them is. The effective distance is defined as follows:*

$$eff_{dis}(i, j) = (1 - \log F_{ji}) \geq 1, \tag{4}$$

where F_{ji} represents the information flow ratio from node v_j to node v_i—$F_{ji} = \frac{a_{ji}}{\sum_{i \in \Gamma(j)} a(ji)}$, you can see from the formula, $eff_{dis}(i, j) \neq eff_{dis}(j, i)$.

Definition 5 (Influence). *On one hand, if the node's neighbor has a high influence, the node's influence will also increase. On the other hand, the influence of node on neighbor will decrease with the increase of distance between them [25,38]. Inspired by the inverse-square law and Coulomb's law, the self-influence of nodes can be*

regarded as the mass of particles and the effective distance as the distance between particles. Thus, the influence of nodes on neighbors can be obtained. The influence of the neighbor nodes is defined as follows:

$$Influence(i,j) = \frac{M_i \times M_j}{R_{ij}} = \frac{Weight(V_i) \times Weight(v_j)}{eff_{dis}(i,j)^2}. \tag{5}$$

The sum of the influence of a node on all neighbor nodes as a measure of the importance of this node. The definition is as follows:

$$KDEC(i) = \sum_{j \in \Gamma(i)} Influence(i,j), \tag{6}$$

$$KDEC(i) = \sum_{j \in \Gamma(i)} \frac{Weight(V_i) \times Weight(v_j)}{eff_{dis}(i,j)^2}, \tag{7}$$

where $\Gamma(i)$ represents the neighbor nodes set of the node v_i.

3.2. The KDEC Model

To further illustrate the specific calculation process of the KDEC algorithm, we use a case to explain it in detail. Figure 1 shows a simple network graph G with 14 nodes and 18 edges. Take the influence of calculating node v_7 in the network as an example.

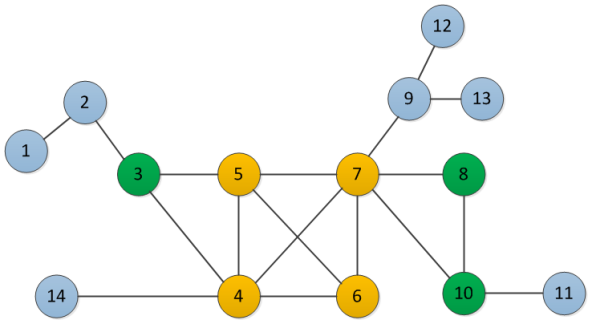


Figure 1. A toy network figure.

Calculate the weight of the node itself. The product of the position of nodes in the network and the local influence of the node is used as a measure of the weight of the node itself. According to Equations (1) and (2), the basic properties of the node can be calculated, as shown in Table 1. The number of kernel cores of the node v_7 is 3 and the degree is 6, so its initial weight can be obtained Equation (3): $Weight(v_7) = Ks(v_7) \times d(v_7) = 3 \times 6 = 18$. Similarly, the initial weights of other nodes can be obtained.

Table 1. Properties of each node in the toy network.

node	1	2	3	4	5	14	6	7	8	9	10	11	12	13
K-shell	1	1	2	3	3	1	3	3	2	1	2	2	1	1
Degree	1	2	3	5	4	1	3	6	2	3	3	1	1	1

Calculate the effective distance. The effective distance between nodes depends on the closeness between nodes and neighbors. The greater the probability that neighbor nodes are affected and the more information obtained, the shorter the effective distance between nodes will be. Node v_7 has 6 nearest neighbor nodes and 5 next-nearest neighbor nodes, respectively taking the nearest

neighbor node v_4 and the next-nearest neighbor node v_{14} as examples. According to Equation (4), effective distance is calculated as follows.

$$eff_{dis}(7,4) = (1 - \log F_{4,7}) = 1.69897.$$

When calculating the distance between the node and the next neighbor node, the effective distance should be added.

$$eff_{dis}(7,14) = eff_{dis}(7,4) + eff_{dis}(4,14) = 1.6989 + 1 = 2.6989.$$

Calculate the influence on neighbor nodes. According to the "three-degree separation" theory of human social behavior communication [39], when considering the contribution of neighbor nodes to node centrality, there is a saturation effect. The best node centrality can be obtained when considering the nearest neighbors and the next-nearest neighbors. When considering the next-next-nearest neighbors, the performance of node centrality no longer improves or even deteriorates. Therefore, the KDEC method only considers the nearest neighbors and the next-nearest neighbors when calculating the effect on neighbor nodes, which greatly reduces the time complexity of calculation.

Taking the nearest neighbor node v_4 as an example, we can calculate the influence of node v_7 on it according to Equation (5).

$$Influence(7,4) = \frac{Weight(v_7) \times Weight(v_4)}{n \times eff_{dis}(7,4)^2} = 6.68135,$$

where n is the number of nodes in the network. Similarly, taking the next-nearest neighbor node v_{14} as an example, we can calculate the influence of node v_7 on it according to Equation (5):

$$Influence(7,14) = \frac{Weight(v_7) \times Weight(v_{14})}{n \times eff_{dis}(7,14)^2} = 0.17650.$$

According to this method, the influence of the node v_7 on all the nearest neighbors and the next-nearest neighbors can be calculated.

Calculate the total influence of the node. The total influence of nodes in the network refers to the sum of the influence of nodes on the nearest neighbor nodes and the next-nearest neighbor nodes. According to Equation (6), the total influence of node v_7 in the network is

$$KDEC(v_7) = \sum_{j \in \Gamma(i)} Influence(7,j) + \sum_{k \in \Gamma(j), k \neq i} Influence(7,k) = 28.7212,$$

where v_j and v_k respectively represent the nearest neighbor nodes and the next-nearest neighbor nodes of the node v_i. Finally, we can get the influence ranking results of all nodes, as shown in Table 2:

Table 2. Influence ranking of toy network nodes.

node	7	4	5	6	3	10	8	9	2	14	11	13	12	1
Rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14

3.3. The KDEC Algorithm

Algorithm 1 gives the pseudo-code for KDEC. The input to the algorithm is a network containing the node V and the edge E. First of all, the initial weight of the node is obtained by Equations (1)–(3). Next, the effective distance between the node and all the nearest neighbors and the next-nearest neighbors are obtained by Equation (4). Then, the initial weight of nodes and the effective distance between nodes are integrated, the influence ability of nodes on their neighbors is calculated by Equation (5). Finally, the influence ability of nodes on all neighbors is obtained through Equation (6), which is used as the index of nodes' influence ability in the network, and the nodes are ranked.

Algorithm 1: KDEC.

Input:

$G = (V, E)$;

network G with N nodes and E edges;

Output:

traverse the nodes;

// The weight of nodes.

for each node v in G=(V,E) **do**

  Get degree of v

  Get k_core of v

  //Calculate the interaction with the nearest neighbor nodes.

  for node u in G.neighbors of node v **do**

    compute Wight(v) using (3)

    compute Wight (u) using (3)

    compute $eff_{dis}(v, u)$ using (4)

    compute influence (v,u) using (5)

    // Calculate the interaction with the next-nearest neighbor node.

    for node w in G.neighbors of node u **do**

      compute Wight(v) using (3)

      compute Wight (w) using (3)

      compute $eff_{dis}(v, w)$ using (4)

      compute influence (v,w) using (5)

    end for

  end for

end for

//The interact influnce of all neighbor nodes.

compute KDEC(v)using(6)

return KDEC(v)

3.4. Time Complexity

The computational complexity of KDEC algorithms has three main components. The first step is to calculate the weight of the node itself. The degree and k-score of nodes in the network need to be identified, and the time complexity of the calculation is $O(|E|)$. In the second step, the effective distance between nodes and their neighbors is calculated. It is necessary to traverse all nodes in the network and their nearest neighbors and the next-nearest neighbors, then calculate their effective distance. The time complexity of this step is $O(N < k >^2)$, where $< k >$ is the average degree of the network and N is the number of network nodes. In the third and fourth steps, the influence of the node on all neighbor nodes is calculated. It does not affect the time complexity of the algorithm, so it can be ignored. Therefore, the computational complexity of the KDEC algorithm is $O(N < k >^2 + |E|)$.

4. Experimental Evaluation

4.1. Real-World Network Data Sets

Six different real networks were selected in the experiment, and the basic statistics are shown in Table 3. (i) **Ca-AstroPh**. Ca-AstroPh is a collaboration graph of authors of scientific papers from the arXiv's Astrophysics (Astro-ph) section; (ii) **Arenas-email**. This is the email communication network at the University Rovira I Virgili in Tarragona in the south of Catalonia in Spain. Nodes are users and each edge represents that at least one email was sent; (iii) **Friendships**. This network contains friendships among users of the website hamsterster.com; (iv) **As2000010**. This is an astatic network of interconnected autonomous systems on the Internet. Nodes are autonomous systems (AS) and edges represent communication; (v) **Bio-dmela**. This network is a collection of biological networks. The nodes are proteins and the edges represent protein–protein interactions; (vi) **Web-spam**.

Network data set provided by Purdue university network repository, which has 4767 nodes and 37,375 edges. These networks can be download from KONECT (http://konect.uni-koblenz.de/networks/).

Table 3. The statistics of six real-world complex networks: Node number $|V|$, edge number $|E|$, average degree $< K >$, maximum degree K_{max}, and clustering coefficient $< CC >$.

Data Sets	$\|V\|$	$\|E\|$	$< K >$	K_{max}	$< CC >$
Arenas-email	1133	5451	9.62	71	0.2202
Friendship	1858	12,534	5.76	85	0.167
As2000010	6474	12,572	3.883	1458	0.252
Bio-dmela	7393	25,569	6.916	17	0.5706
Web-spam	4767	37,375	15.681	477	0.2859
Ca-astroph	18,771	198,050	21.34	236	0.677

4.2. Algorithm Description

In the experiment, 6 ranking algorithms are selected for comparison. The comparison algorithm is described as follows:

CC (Closeness Centrality): Closeness centrality determines the influential nodes based on global information. It is necessary to calculate the relative distance between each pair of nodes to determine the centrality of nodes.

EC (Eigenvector Centrality): Eigenvector centrality considers that the importance of nodes depends not only on the number (degree) of adjacent nodes, but also on the importance of adjacent nodes. It evaluates the impact of a node based on information about other nodes that are disconnected from the node.

HITS: HITS focuses on nodes in the ranking network and uses different metrics together to evaluate the influence of nodes. Two attributes, hub value and authority value, are assigned to each node to evaluate the impact of the node. The authority value measures the original creativity of nodes to information; and the hub value reflects the role of nodes in information transmission. They interact and converge iteratively.

H-index: The algorithm is mainly used to evaluate a scholar's academic achievements. The calculation method of H in the index of H is that there are at least H citations of H articles published by a scholar. The higher H-index value indicates the greater influence of this node.

PageRank: PageRank ranks pages according to link structure. It argues that the impact of a web page depends on the number and quality of other pages pointing to it. If a page has many high-quality pages pointing to it, the quality of the page will be high. It works well in directed networks, but not in undirected networks.

ProfitLeader: ProfitLeader [40] is the latest of these comparison algorithms and was proposed in May 2018. The algorithm quantifies the profit provided by nodes by ordering the influential nodes in the network. The calculation process is relatively simple and suitable for large networks.

4.3. Evaluate Metric

4.3.1. SIR Model

There are two methods to evaluate the accuracy of the influential nodes identification algorithm. The method based on communication dynamics is widely used by scholars at home and abroad to verify the accuracy of the ranking index in identifying influential nodes in communication [23,41]. In our experiments, the ranking results were evaluated using the SIR (Susceptible-Infected-Recover) [42] model. Now, the SIR model mainly includes nodes of three states: S (Susceptible to infection) means that the node is healthy, susceptible to being infected by others, and unlikely to infect others; I (Infected) state means that the individual is in an infected state and will infect other nodes, but may also be recovered to the R state; R (Recover) state means the node has been infected by another node but recovered.

S(t), I(t), and R(t) are the number of nodes in susceptible, infected, and recovered states, and S(t), I(t), and R(t) are the proportion of nodes in susceptible, infected, and recovered states, respectively, so

$$\begin{cases} \frac{ds(t)}{d(t)} = -\beta s(t)i(t) \\ \frac{di(t)}{d(t)} = \beta s(t)i(t) - \beta i(t) \\ \frac{dr(t)}{d(t)} = \beta i(t) \end{cases} . \tag{8}$$

The SIR model evaluates the performance and effectiveness of the algorithm through multiple iterative experiments. A node is selected as the seed node in each process, and the average number of infected and recovered nodes after t-step infection is taken as the propagation capacity of the node, denoted as F(t). The SIR propagation influence K_i of the node V_i is defined as

$$K_i = F(t) = \frac{1}{N_{ite}}(nI + nR), \tag{9}$$

where N_{ite} refers to the number of iterations. In our experiment, we set $N_{ite} = 1000$, and nI and nR refer to the number of infected and recovered nodes, respectively.

4.3.2. Kendall Correlation Coefficient

In this paper, the Kendall coefficient [43] is introduced to measure the correlation between the ranking results of these comparison algorithms and the SIR model. The Kendall coefficient is used to illustrate the rank correlation of variables. Generally, one of the data is required to be ordered, because the correlation coefficient in the ordered case can reflect the closeness of two sequences. It assumes that the two sequences are associated with the same number of nodes n, expressed as $X = (x_1, x_2, ..., x_n), Y = (y_1, y_2..., y_n)$. If two elements of the sequence are consistent, if $x_i > x_j$ and $y_i > y_j$, or if $x_i < x_j$ and $y_i < y_j$, then any pair of binary group are considered to be consistent. On the contrary, if $x_i > x_j$ and $y_i < y_j$, or if $x_i < x_j$ and $y_i > y_j$, then they are not consistent. If $x_i = x_j$ or $y_i = y_j$, then it is neither consistent nor inconsistent. Kendall tau coefficient is defined as

$$\tau(X, Y) = \frac{n_c - n_d}{0.5n(n-1)}, \tag{10}$$

where n_c and n_d represent the number of consistent and inconsistent binary groups. It can be noticed that its value is consistent with the ordered list. The higher the τ value, the more accurate the ranked list generated by this method.

4.4. Evaluation Analysis

This paper firstly uses the toy network diagram as an example to verify the validity of KDEC. In Figure 1, the color of the nodes represents different k_core numbers, the gray represents a k_core number of 1, the green represents a k_core number of 2, and the yellow represents a k_core number of 3. The KDEC algorithm is applied to identify the influential nodes in the toy network, and the results are compared with the CC, EC, PageRank, HITS, Hindex, and ProfitLeader algorithms and SIR propagation detection results. The results are shown in Table 4. As can be seen from the table, the ranking result of KDEC algorithm is consistent with the detection sequence of the SIR model, and its effect is better than other ranking algorithms.

Table 4. Ranking results in toy network by KDEC and other comparison algorithms.

Rank	CC	EC	Hits	H-index	PR	PL	KDEC	KDEC Value	SIR	SIR Value
1	7	7	7	4	7	7	7	28.7212	7	2.907
2	4	4	4	5	4	4	4	21.1858	4	2.494
3	5	5	5	6	9	5	5	16.6559	5	2.455
4	9	6	6	7	5	9	6	11.7006	6	2.31
5	3	9	3	3	10	10	3	6.9794	3	2.03
6	6	10	9	8	3	3	10	5.3482	10	1.93
7	10	3	10	10	6	6	8	3.4368	9	1.852
8	2	8	8	2	2	2	9	2.5331	8	1.843
9	1	14	14	9	1	1	2	0.9289	2	1.54
10	14	13	2	1	13	11	14	0.6624	14	1.468

4.4.1. Efficiency Analysis

In the propagation verification of SIR, if the value of λ is large, seed nodes may rapidly infect the whole network [23], so it is difficult to distinguish the propagation influence of different nodes. In this experiment, we used a relatively small β, specifically, $\beta \in [0.01, 0.1]$; recovery probability $\gamma = 1$ [12]; and time step t = 1000.

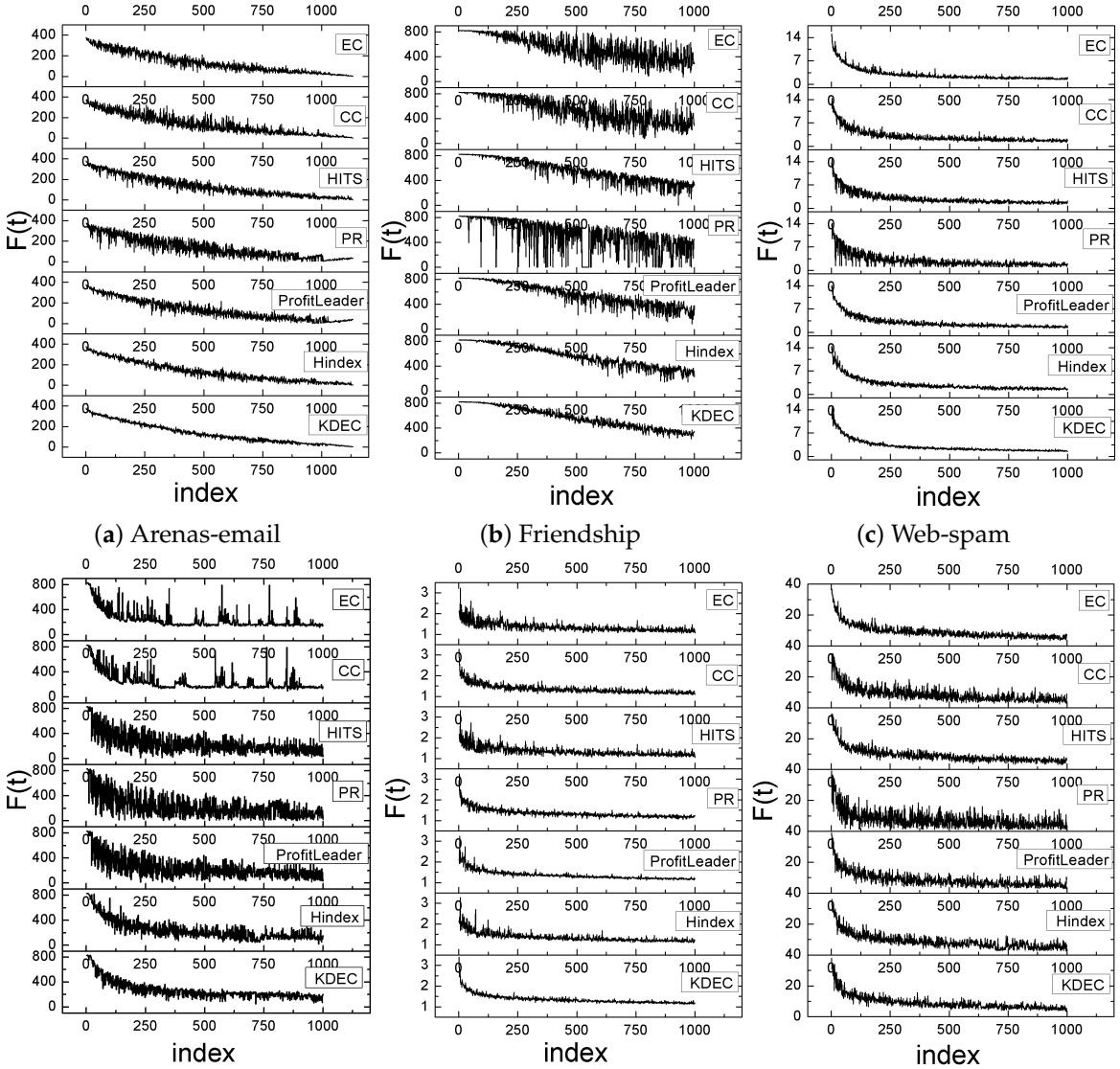
In order to further evaluate the performance of the algorithm, each node was taken as a seed node, and the average number of infected nodes of each seed node was calculated through more than 1000 runs. The index refers to the influence of nodes in descending order, F(t) is the number of infected and recovered nodes at time t, that is, the number of infected nodes corresponding to each node. All infected nodes can be observed from a global perspective. In the SIR model, influential nodes can infect more nodes. Therefore, the graph generated by the infection curve of influential nodes identified by a good algorithm should decrease from left to right. The smoother the curve is, the smaller the fluctuation is, and the number of infected nodes decreases with the increase of index, indicating that the ranking result is more reasonable.

As shown in Figure 2, the KDEC method has the best infection effect compared with the other algorithms in the six networks. However, in the process of infection, the performance of other algorithms is not stable on different data sets, the infection curve has many peaks and troughs, which indicates that the algorithm has limitations in application and deviations from the real ranking results. For example, the EC performs well in Arenas-email (Figure 2a), As2000010 (Figure 2c) and Ca-astroph (Figure 2f) networks, but poorly in other networks. The HIT performs better in Arenas-email, Friendships (Figure 2b) and Ca-astroph networks, but performs worse in the remaining networks. ProfitLeader and Hindex have poor performance in Bio-dmela (Figure 2d) and Web-spam (Figure 2e) network respectively, and have good performance in other networks, but not as good as KDEC. PageRank only performs well in web-spam network. The CC performed poorly in all networks. It can be seen from the experimental results that compared with other algorithms, the method proposed in this paper has a better infection effect on most networks, indicating that it is more accurate and effective in identifying influential nodes.

4.4.2. Infectivity Analysis

In order to more comprehensively compare the performance of various algorithms for identifying influential nodes, we compare the influence of the top-10 nodes that are discriminatively selected by KDEC and other comparison algorithms. Since the same node has the same number of infected nodes, we remove the same node from the sorting results of the two algorithms and select the top-10 nodes. Unlike the previous experiment where a single node was used as a seed node, this experiments all 10 nodes as a seed node, to begin with. The cumulative number of infected nodes increases with the increase of time step, finally obtaining the stable value after several time steps. Set the propagation threshold [44] as $\beta = < k > / < k^2 >$, where $< k >$ is the average degree of nodes in the network, $< k^2 >$ is the average degree of the next nearest neighbors, and the recovery probability is

$\gamma = 1$. F(t) refers to the number of infected and recovered nodes at time t, which varies from 1 to 20. In Figure 3, the red curve represents the KDEC algorithm infection curve, and the black curve represents the comparison algorithm infection curve. The higher the infection curve is above the coordinate axis, the more nodes will be infected at time step t, and the better the performance of the algorithm will be.



(**a**) Arenas-email (**b**) Friendship (**c**) Web-spam

(**d**) As2000010 (**e**) Bio-dmela (**f**) Ca-astroph

Figure 2. Infection effect of influential nodes identified by each comparison algorithm.

From the sub-graph of Figure 3, the infection effect of the top-10 nodes can be directly seen. In general, the KDEC method achieves better infection performance than other algorithms. Specifically, KDEC achieved the best performance on Web-spam (Figure 3c), As2000010 (Figure 3d), and Bio-dlema networks (Figure 3e), with the infection curve always above the axis. In Arenas-email (Figure 3a) and Friendships network (Figure 3b), the effect of KDEC algorithm is similar to that of ProfitLeader and Hindex, and KDEC is better than that of other algorithms. In Ca-astroph network (Figure 3f), the infection effect of KDEC was significantly better than that of EC, CC, HITS, PageRank, and ProfitLeader, and there was little difference in the number of nodes infected with Hindex. From the experimental data of six real networks, it can be concluded that the top-10 different nodes obtained by KDEC have better infection effect than other algorithms, and it has obvious advantages in propagation.

4.4.3. Kendall Coefficient Analysis

To better distinguish and verify the effectiveness and reliability of the KDEC method, Kendall was used to compare the accuracy of each comparison algorithm. Similarly, each comparison algorithm is applied to several real-world networks—such as Arenas-email, Friendships, Bio-dmela, As2000010, and Web-spam—and the ranking results are compared with the results of the SIR model to calculate its Kendall correlation coefficient. The calculation results are shown in Figure 4.

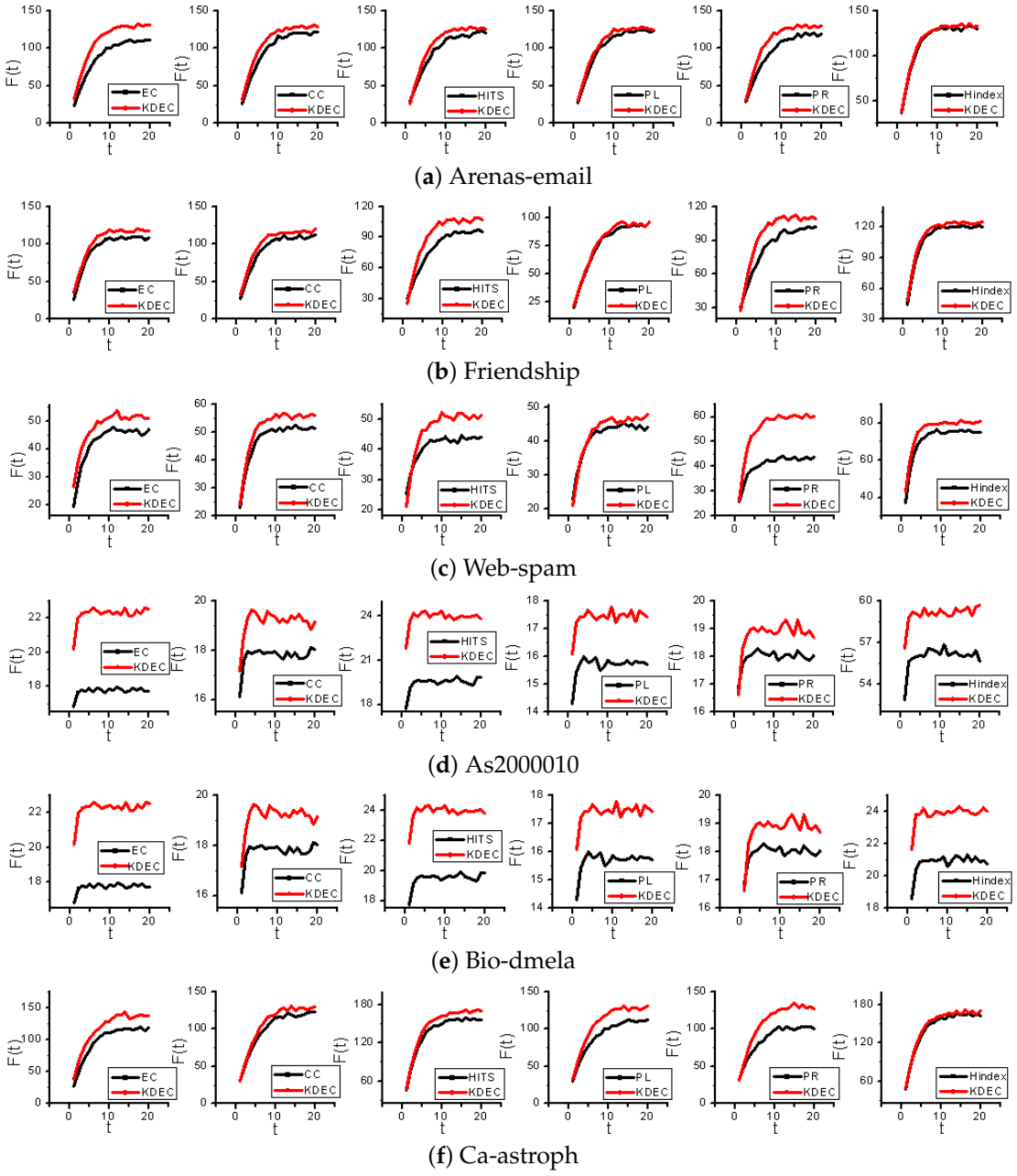


Figure 3. Propagation effects of the top-10 different nodes of KDEC and other comparison algorithms.

Our algorithm achieves the highest Kendall coefficient in Arenas-email, Friendship, As2000010, Web-spam and Ca-astroph networks, and our results reach above 0.9 in Arenas-email and Friendships; while in the other three networks, it stays between 0.75 and 0.9. In Bio-dmela network, KDEC's Kendall coefficient is not as high as that of the ProfitLeader algorithm. However, there is little difference between the Kendall coefficient of the two algorithms. The Kendall coefficient of the ProfitLeader algorithm is 0.8985 and that of the KDEC algorithm is 0.8960. This shows that the result sequence

ranked by KDEC is basically consistent with the real SIR model. It shows that the algorithm KDEC proposed in this paper can accurately identify the influential nodes in the network.

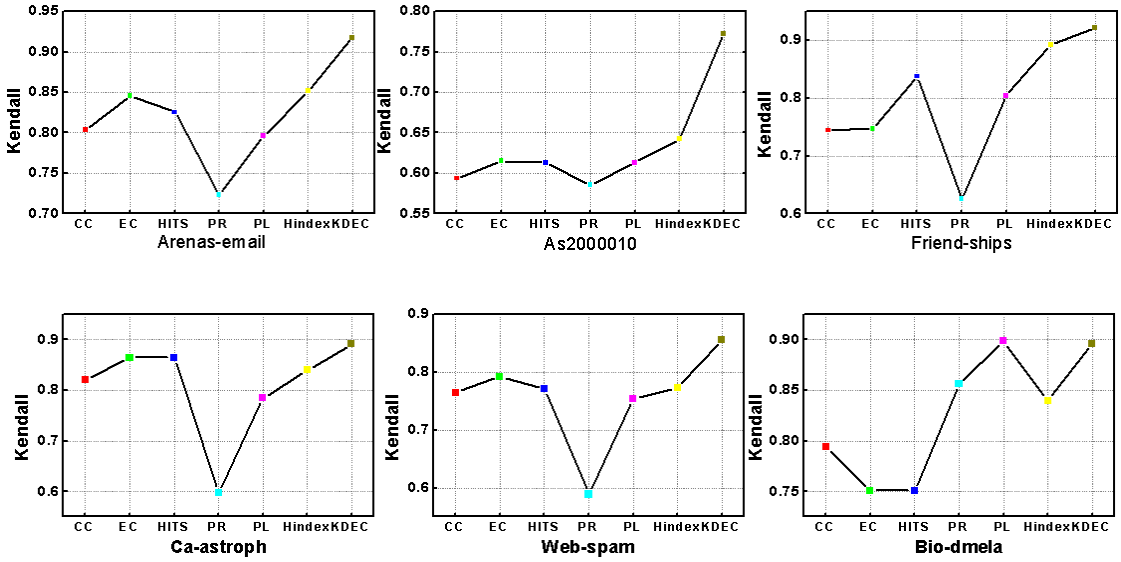


Figure 4. The results of all comparison algorithms on each real-world network, obtained through 1000 independent runs.

5. Conclusions

In this paper, we change the traditional method of distance evaluation between nodes, conduct in-depth research on the effective distance between nodes, and comprehensively evaluate the influence ability of nodes in the network by combining the position attribute of nodes in the network and the interaction force on neighboring nodes. In order to verify the effectiveness of this ranking method, we selected six real networks with different network structure attributes and carried out experimental verification from three aspects, namely, node infection capacity, transmission effectiveness, and consistency with the results of the standard SIR model. From the experimental results, the proposed KDEC method has obvious advantages over the classical sorting method and the recently proposed ProfitLeader ranking method in node importance sorting. However, our method selected node degree value and K_core as initial weights of nodes, the product of this in the computing time complexity is a little high, and in this article, when considering the influence of the neighbor nodes, we only got the second-order neighbors. The node importance evaluation accuracy needs to be improved, they are the next steps we need to take to improve the proposed algorithm.

Author Contributions: Conceptualization, J.S. and B.W.; methodology, J.Z.; supervision, J.S. and B.W.; validation, J.Z., J.D., J.H. and L.C.; writing, J.S. and J.Z.

Funding: This study was supported by the National Science and Technology Major Project of China [2017ZX06002005].

Acknowledgments: We thank the editor and reviewers for their thorough reviews, thoughtful comments, and constructive suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Erdös, P.; Rényi, A. Publication of the Mathematical Institute of the Hungarian Academy of Sciences. 1960. Available online: http://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?ReferenceID=1448571 (accessed on 8 October 2019).
2. Barrat, A.; Barthelemy, M.; Vespignani, A. *Dynamical Processes on Complex Networks*; Cambridge University Press: New York, NY, USA, 2008.

3. Dorogovtsev, S.N.; Goltsev, A.V.; Mendes, J.F. Critical phenomena in complex networks. *Rev. Mod. Phys.* **2008**, *80*, 1275. [CrossRef]
4. Raven, B. *The Bases of Social Power*; University of Michigan: Oxford, UK, 1959; pp. 150–167.
5. Zhou, T.; Liu, J.G.; Bai, W.J. Behaviors of susceptible-infected epidemics on scale-free networks with identical infectivity. *Phys. Rev.* **2006**, *74*, 056109. [CrossRef]
6. Pastorsatorras, R.; Vespignani, A. Epidemic Spreading in Scale-Free Networks. *Phys. Rev. Lett.* **2000**, *86*, 3200–3203. [CrossRef]
7. Shah, D.; Zaman, T.R. Detecting sources of computer viruses in networks: Theory and experiment. In Proceedings of the 2010 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, New York, NY, USA, 14–18 June 2010.
8. Tao, L.; Liu, X.; Li, H. An Immune-Based Model for Computer Virus Detection. *Lect. Notes Comput. Sci.* **2005**, *3810*, 59–71.
9. Doer, B.; Fouz, M.; Friedrich, T. Why rumors spread so quickly in social networks. *Commun. ACM* **2012**, *55*, 70. [CrossRef]
10. Moreno, Y.; Nekovee, M.; Pacheco, A.F. Dynamics of rumor spreading in complex networks. *Phys. Rev.* **2004**, *69*, 066130. [CrossRef]
11. Yan, Y.; Qian, Y.; Sharif, H. A Survey on Smart Grid Communication Infrastructures: Motivations, Requirements and Challenges. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 5–20. [CrossRef]
12. Liu, J.; Lin, J.; Guo, Q.; Zhou, T. Locating influential nodes via dynamics-sensitive centrality. *Sci. Rep.* **2016**, *6*, 21380. [CrossRef]
13. Brockmann, D.; Helbing, D. The hidden geometry of complex, network—Driven contagion phenomena. *Science* **2013**, *342*, 1337–1342. [CrossRef]
14. Fei, L.; Zhang, Q.; Deng, Y. Identifying influential nodes in complex networks based on the inverse—Square law. *Phys. Stat. Mech. Appl.* **2018**, *512*, 1044–1059. [CrossRef]
15. Ma, L.; Ma, C.; Zhang, H.; Wang, B. Identifying influential spreaders in complex networks based on gravity formula. *Phys. Stat. Mech. Appl.* **2016**, *451*, 205–212. [CrossRef]
16. Helbing, D. Globally networked risks and how to respond. *Nature* **2013**, *497*, 51–59. [CrossRef]
17. Gao, C.; Lan, X.; Zhang, X.; Deng, Y. A bio-inspired methodology of identifying influential nodes in complex networks. *PLoS ONE* **2013**, *8*, e66732. [CrossRef]
18. Ttcher, L.; Woolley-Meza, O.; Goles, E. Connectivity disruption sparks explosive epidemic spreading. *Phys. Rev.* **2016**, *93*, 042315.
19. Gao, L.; Song, J.; Nie, F.; Zou, F.; Sebe, N.; Shen, H.T. Graph-without-Cut: An ideal graph learning for image segmentation. In Proceedings of the Thirtieth Aaai Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
20. Song, J.; Gao, L.; Zou, F.; Yan, Y.; Sebea, N. Deep and fast: Deep learning hashing with semi-supervised graph construction. *Image Vis. Comput.* **2016**, *55*, 101–108. [CrossRef]
21. Lv, L.; Chen, D.; Ren, X.L.; Zhang, Q.M.; Zhang, Y.C.; Zhou, T. Vital nodes identification in complex networks. *Phys. Rep.* **2016**, *650*, 1–63.
22. Bonacich, P. Factoring and weighting approaches to status scores and clique identification. *J. Math. Cal Sociol.* **1972**, *2*, 113–120. [CrossRef]
23. Kitsak, M.; Gallos, L.K.; Havlin, S.; Liljeros, F.; Muchnik, L.; Stanley, H.E.; Makse, H.A. Identification of influential spreaders in complex networks. *Nat. Phys.* **2010**, *6*, 888. [CrossRef]
24. Zhao, S.X.; Rousseau, R.; Ye, F.Y. h-Degree as a basic measure in weighted networks. *J. Inf.* **2011**, *5*, 668–677. [CrossRef]
25. Hage, P.; Harary, F. Eccentricity and centrality in networks. *Soc. Netw.* **1995**, *17*, 57–63. [CrossRef]
26. Sabidussi, G. The centrality index of a graph. *Psychometrika* **1966**, *31*, 581–603. [CrossRef] [PubMed]
27. Katz, L. A new status index derived from sociometric analysis. *Psychometrika* **1953**, *18*, 39–43. [CrossRef]
28. Freeman, L.C. Centrality in social networks conceptual clarification. *Soc. Netw.* **1978**, *1*, 215–239. [CrossRef]
29. Yan, G.; Zhou, T.; Hu, B.; Fu, Z.; Wang, B. Efficient routing on complex networks. *Phys. Rev.* **2006**, *73*, 046108. [CrossRef] [PubMed]
30. Zeng, A.; Zhang, C. Ranking spreaders by decomposing complex networks. *Phys. Lett.* **2013**, *377*, 1031–1035. [CrossRef]

31. Estrada, E.; Rodriguez–Velazquez, J.A. Subgraph centrality in complex networks. *Phys. Rev.* **2005**, *71*, 056103. [CrossRef]
32. Poulin, R.; Boily, M.C.; Masse, B.R. Dynamical systems to define centrality in social networks. *Soc. Netw.* **2000**, *22*, 187–220. [CrossRef]
33. Brin, S.; Page, L. The anatomy of a large–scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **1998**, *30*, 107–117. [CrossRef]
34. Lv, L.; Zhang, Y.; Yeung, C.H.; Zhou, T. Leaders in social networks, the delicious case. *PLoS ONE* **2011**, *6*, e21202.
35. Kleinberg, J.M. Authoritative sources in a hyperlinked environment. *J. ACM* **1999**, *46*, 604–632. [CrossRef]
36. Chen, D.B.; Gao, H.; Lv, L.; Zhou, T. Identifying influential nodes in largescale directed networks: The role 410 of clustering. *PLoS ONE* **2013**, *8*, e77455.
37. Barabasi, A.; Albert, R. Emergence of scaling in random networks. *Science* **1999**, *286*, 509–512. [CrossRef] [PubMed]
38. Kang, B.; Chhipi-Shrestha, G.; Deng, Y.; Hewageb, K.; Sadiq, R. Stable strategies analysis based on the utility of Z-number in the evolutionary games. *Appl. Math. Comput.* **2018**, *324*, 202–217. [CrossRef]
39. Christakis, N.A.; Fowler, J.H. Social contagion theory: Examining dynamic social networks and human behavior. *Stat. Med.* **2013**, *32*, 556–577. [CrossRef] [PubMed]
40. Yu, Z.; Shao, J.; Yang, Q.; Sun, Z. Profitleader: Identifying leaders in networks with profit capacity. *World Wide Web* **2019**, *22*, 533–553. [CrossRef]
41. Hu, Y.; Ji, S.; Feng, L.; Jin, Y. Quantify and maximise global viral influence through local network information. *arXiv* **2015**, arXiv:1509.03484.
42. Anderson, R.M.; Anderson, B.; May, R.M. *Infectious Diseases of Humans: Dynamics and Control*; Oxford University Press: New York, NY, USA, 1991.
43. Kendall, M.G. A new measure of rank correlation. *Biometrika* **1938**, *30*, 81–93. [CrossRef]
44. Moreno, Y.; Pastor-Satorras, R.; Vespignani, A. Epidemic outbreaks in complex heterogeneous networks. *Eur. Phys. J.* **2002**, *26*, 521–529. [CrossRef]