

Identifying influential spreaders in complex networks based on density entropy and community structure

Zhan Su(苏湛), Lei Chen(陈磊), Jun Ai(艾均), Yu-Yu Zheng(郑雨语), and Na Bie(别娜)
Chin. Phys. B, 2024, 33 (**5**): 058901. DOI: 10.1088/1674-1056/ad20d6
Citation: Haoran Yan(严浩然), Jiaqi Yan(颜家琦), Lianchun Yu(俞连春), and Yu-Feng Shao(邵玉峰)
What follows is a list of articles you may be interested in

A multilayer network diffusion-based model for reviewer recommendation

Yiwei Huang(黄羿炜), Shuqi Xu(徐舒琪), Shimin Cai(蔡世民), and Linyuan Lü(吕琳媛)
Chin. Phys. B, 2024, 33 (**3**): 038901. DOI: 10.1088/1674-1056/ad181d

Source localization in signed networks with effective distance

Zhi-Wei Ma(马志伟), Lei Sun(孙蕾), Zhi-Guo Ding(丁智国), Yi-Zhen Huang(黄宜真), and Zhao-Long Hu(胡兆龙)
Chin. Phys. B, 2024, 33 (**2**): 028902. DOI: 10.1088/1674-1056/ad1482

Identifying multiple influential spreaders in complex networks based on spectral graph theory

Dong-Xu Cui(崔东旭), Jia-Lin He(何嘉林), Zi-Fei Xiao(肖子飞), and Wei-Ping Ren(任卫平)
Chin. Phys. B, 2023, 32 (**9**): 098904. DOI: 10.1088/1674-1056/acac0b

Identification of key recovering node for spatial networks

Zijian Yan(严子健), Yongxiang Xia(夏永祥), Lijun Guo(郭丽君), Lingzhe Zhu(祝令哲), Yuanyuan Liang(梁圆圆), and Haicheng Tu(涂海程)
Chin. Phys. B, 2023, 32 (**6**): 068901. DOI: 10.1088/1674-1056/acb75f

AG-GATCN: A novel method for predicting essential proteins

Peishi Yang(杨培实), Pengli Lu(卢鹏丽), and Teng Zhang(张腾)
Chin. Phys. B, 2023, 32 (**5**): 058902. DOI: 10.1088/1674-1056/acb9f9

Identifying influential spreaders in complex networks based on density entropy and community structure

Zhan Su(苏湛), Lei Chen(陈磊)†, Jun Ai(艾均), Yu-Yu Zheng(郑雨语), and Na Bie(别娜)

School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, *China*

(Received 17 October 2023; revised manuscript received 17 January 2024; accepted manuscript online 22 January 2024)

In recent years, exploring the relationship between community structure and node centrality in complex networks has gained significant attention from researchers, given its fundamental theoretical significance and practical implications. To address the impact of network communities on target nodes and effectively identify highly influential nodes with strong propagation capabilities, this paper proposes a novel influential spreaders identification algorithm based on density entropy and community structure (DECS). The proposed method initially integrates a community detection algorithm to obtain the community partition results of the networks. It then comprehensively considers the internal and external density entropies and degree centrality of the target node to evaluate its influence. Experimental validation is conducted on eight networks of varying sizes through susceptible–infected–recovered (SIR) propagation experiments and network static attack experiments. The experimental results demonstrate that the proposed method outperforms five other node centrality methods under the same comparative conditions, particularly in terms of information spreading capability, thereby enhancing the accurate identification of critical nodes in networks.

Keywords: complex networks, influential spreaders, propagation model, static attack

PACS: 89.75.Fb **DOI:** 10.1088/1674-1056/ad20d6

1. Introduction

In complex network models, centrality measurements serve to quantify the importance of nodes that significantly influence the overall stability of the structure. Considering that many real-world complex systems can be effectively represented by complex network models, the study and analysis of centrality measurements in such networks are of crucial theoretical significance.

Most real-life complex systems can be regarded as complex network models, where protecting key nodes in the network becomes particularly important. For instance, in traffic networks, congested road sections are more likely to cause traffic jams. In neural networks, neurons transmitting critical signals are more likely to contribute to the occurrence of paralyzing diseases, and the failure of critical substations in power grids can result in widespread power outages.[1–3] Consequently, the exploration of node centrality in complex networks is indispensable and offers substantial practical value.

Complex networks have many classic centrality algorithms, such as degree centrality,[4] H-index centrality,[5] betweenness centrality,[6] closeness centrality,[7] eigenvector centrality,[8] PageRank,[9] and K-shell.[10] These methods measure the importance of nodes from different perspectives, including node neighbors, global paths, eigenvectors, and network positions.

In recent years, there has been a surge in the innovative application of deep learning methods to investigate node influence by both domestic and international researchers. Rezaei *et al.*[11] introduced a machine learning data-driven method for identifying node importance. This method uses a support vector machine model to train on a subset of the network graph and predict the remaining nodes. Additionally, Zhao *et al.*[12] proposed a deep learning model called InfGCN based on graph convolutional networks to identify optimal nodes in complex networks. Machine learning-based node centrality methods require a large amount of training data to obtain accurate predictive results. Their computational time complexity is high, and the interpretability of node centrality methods is poor.

In order to accurately evaluate the importance of nodes in a network, it is necessary to consider the relative distance and connectivity strength between nodes. Li *et al.*[13] introduced a localized version of the gravity model by introducing a cutoff radius to reduce cumulative errors caused by distance interactions and decrease computational complexity. Yang *et al.*[14] proposed an improved gravity centrality measurement method called KSGC based on the K-shell algorithm. Li *et al.*[15] introduced a novel gravity model that incorporates the multi-characteristics of nodes called MCGM. The node centrality methods based on the gravity model are capable of detecting influential nodes. Nevertheless, these methods are susceptible to the influence of node degree values and exhibit high computational time complexity in large-scale networks.

To consider the impact of structural hole nodes on network centrality, numerous researchers have conducted relevant studies. Zhao *et al.*[16] proposed a ranking algorithm called SHKS based on the adaptable fusion of the structural hole

†Corresponding author. E-mail: 213330655@st.usst.edu.cn

© 2024 Chinese Physical Society and IOP Publishing Ltd

(SH) and K-shell. With this method, a node's neighbors and second-order neighbors are taken into account in addition to its K-shell and SH-based centrality. Han *et al.*[17] proposed a key node ranking method based on structural holes, incorporating a ListNet-based ranking learning approach. This method simultaneously utilizes seven metrics as the features of structural hole nodes to comprehensively rank nodes in the network. Jiang *et al.*[18] introduced an innovative method termed EGMS, which thoroughly analyzes the structural hole features and K-shell centrality of nodes, taking into account both local topology and global location information. These methods have not considered the influence of network communities on node centrality.

Additionally, the triangle structure formed by nodes in complex networks is also utilized in the study of node centrality. Han *et al.*[19] proposed a node influence measure index model known as LTC, which is founded on a triangular structure between a node and its neighbor nodes. The model takes into consideration not only the triangle structure among nodes, but also the degree of the surrounding neighbor nodes. Ma *et al.*[20] introduced a local triangle structure centrality method called LTSC, which is based on the degree and the neighbors' connections to the node. This method combines the triangle structure with the degree information of the neighbors to define the internal and external spreading capabilities of the nodes. Node centrality is calculated by summing the internal and external spreading capabilities of the neighbor nodes of the target node. While these methods based on triangle structure demonstrate a relatively low time complexity, enabling their application to large-scale complex networks, they lack effectiveness in assessing the significance of nodes within communities.

Furthermore, researchers have also applied information entropy to the study of centrality in complex networks. Zhong *et al.*[21] proposed an improved information entropy (IIE) method that considers both the number of neighbors surrounding the target node and its rate of spreading. Wang *et al.*[22] proposed an enhanced method utilizing the iteration factor and information entropy to estimate the propagation capability of each layer of nodes. It not only accurately orders the nodes but also effectively avoids the phenomenon of rich clubs. These methods integrating information entropy can effectively identify key nodes in networks, but their applicability is limited to specific types of networks, and they do not take into consideration the importance of community structures in the network.

On the other hand, the influence of community structure on node centrality in complex networks has gained extensive attention from researchers. Kumar *et al.*[23] proposed a community structure algorithm called CIFR which integrates feature ordering to detect influential nodes using the characteristics of community propagation in the network. Sun *et al.*[24] introduced a new community-based K-shell decomposition algorithm called CKS. The algorithm has high time complexity, and the K-shell indices of the partitioned communities may be the same, which makes it impossible to sort the nodes within the communities. Zhao *et al.*[25] proposed a key node identification method called ICC that comprehensively considers both node proximity and community structure. While these studies consider network communities, they do not account for the influence of internal and external communities on the centrality of target nodes.

Given the influence of community structure in networks on the measurement of node centrality, this paper proposes an influential spreaders identification algorithm based on density entropy and community structure (DECS), which further selects important nodes with strong propagation capabilities. The main contributions of this paper are summarized as follows.

(i) By integrating the community structure with the density entropies of network nodes inside and outside, this paper goes beyond just concentrating on the influence of individual nodes. As an alternative, it thoroughly examines the relationships between nodes and communities in order to assess the centrality of nodes.

(ii) Experiments on eight network datasets show that the nodes selected by the algorithm proposed in this paper are crucial nodes that connect communities, which can effectively identify influential nodes in complex networks in both SIR propagation experiments and network static attack experiments.

2. Preliminaries

In social networks, individuals with a high degree of inter-community connections are capable of efficiently receiving and disseminating information within their respective circles, thereby exerting significant influence. To consider the importance of communities in the process of node centrality measurement, it is crucial to select an appropriate community detection algorithm. In this paper, the Walktrap algorithm will be adopted as the community detection algorithm, which better considers the local connectivity of the network and assigns tightly connected nodes to the same community. Additionally, this algorithm demonstrates excellent robustness and produces steady community detection outcomes with a maximum number of communities. Information entropy, proposed by Shannon, intuitively reflects the uncertainty and amount of information in a variable. It is an unbiased measurement method that is not affected by data distribution. Therefore, this paper applies information entropy to network nodes to reflect their influence within and outside communities. To fully consider node influence within the community, this paper integrates the

Walktrap community detection algorithm with information entropy, thereby introducing a novel method for quantifying the influence of network nodes.

2.1. Walktrap community detection algorithm

The Walktrap algorithm is a community detection algorithm based on random walks. Its main idea is that communities are relatively dense subgraphs, making it easier for random walks to get trapped within a community.[26,27] Nodes within the same community are more likely to be accessed during the random walk process, and the more tightly connected the nodes are, the higher the probability of walking between them. The steps of the algorithm are as follows.

(i) Input the network graph. The nodes and edges in the network are constructed into an undirected graph.

(ii) Calculate similarity. For each pair of nodes, multiple random walks are performed to record the number of common visits between the node pairs, and the similarity is then calculated based on these visit counts.

(iii) Build a similarity matrix. Based on the similarities between pairs of nodes, the algorithm constructs a similarity matrix. This matrix represents the similarity relationships between nodes.

(iv) Hierarchical clustering of communities. The algorithm utilizes a similarity matrix for hierarchical clustering operations. Firstly, each node is treated as an initial community. Subsequently, larger communities are gradually constructed by merging pairs of nodes with higher similarity until the entire network forms a single large community.

(v) Select community partition using modularity Q. In the process of constructing communities, the performance of each community merger is measured using modularity Q. Modularity Q is utilized to assess the quality of community partition and select the best community partition result.

2.2. Information entropy

In the 1940s, Claude Shannon introduced the concept of information entropy by integrating statistical thermodynamics, and provided a mathematical expression for calculating specific information entropy, thereby addressing the quantification issue of information.[28] Information entropy, also known as Shannon entropy, is commonly described using probability theory to measure the uncertainty in a system.

In information theory, assume a discrete random variable X that can take on values $X = \{x_1, x_2, \ldots, x_n\}$ with corresponding probabilities $P = \{p_1, p_2, \ldots, p_n\}$. The Shannon entropy of the random variable X is given by the following mathematical expression:

$$H(X) = -\sum_x P(x) \log_2[P(x)]. \tag{1}$$

The greater the value of the Shannon entropy, the higher the uncertainty of the random variable X, indicating a greater amount of contained information.

2.3. SIR model

The SIR model is widely used to simulate information dissemination and disease propagation. This paper therefore utilizes the susceptible–infected–recovered (SIR) model for conducting node propagation performance experiments.[29] The SIR model classifies nodes into three categories: susceptible nodes, infected nodes, and recovered nodes. Susceptible nodes represent those at risk of infection, and infected nodes represent nodes that are already infected and will infect their neighbor nodes. Meanwhile, recovered nodes indicate nodes that have recovered from the infected state and are no longer infectious.

To eliminate the randomness of the experimental simulation results, this paper introduces the concept of the basic reproduction number (R_0) and designs a strategy to simulate the SIR infection process. The basic reproduction number is defined as the average number of secondary infections caused by primary infections in a completely susceptible population, and it is used to evaluate the infectivity and transmission efficiency of a disease.[30] In this paper, we set the basic reproduction number as 2.3; the infection process is performed only once for direct neighbor nodes at each step.

The number of neighbor nodes that the infected nodes can infect is controlled by R_0, and the state of R_0 in the entire network is continuously tracked during the experimental process. When an infected node infects its neighbor nodes, the number of infected nodes in each step is related to R_0, and in most cases, the number of infected nodes is an integer value.

At the beginning of the propagation experiments, the initial R_0 is 0. Due to R_0 being less than 2.3, an infected node will infect three neighbor nodes. If the average R_0 of the network exceeds 2.3, the infected nodes will only infect two neighbor nodes, which are selected in descending order based on their centrality values. Subsequently, the average R_0 value of the entire network is recalculated. When the number of neighbor nodes infected by an infected node exceeds R_0, the infected node will be labeled as a recovered node. The ratio of the sum of infected and recovered nodes to the total number of nodes is considered as the node's propagation capability. The SIR propagation process ends when it is completed within the given propagation time or when there are no longer any infected nodes in the network. At the end of the propagation process, the larger the proportion of infected and recovered nodes in the network, the stronger the propagation capability of the source node.

3. Proposed method

3.1. DECS

This paper suggests that each node within a community has its internal density and external density. Within the same community, the ratio of the sum of neighbor nodes to the degree of a node in the network represents its internal density. The external density of a node is defined as the ratio of the sum of neighbor nodes connected to other communities to the degree of the node in the network. The influence of a node in sharing or propagating information within and outside the community can be determined through its internal and external densities. The DECS value of a network node can be calculated through the following three steps.

(i) Calculate the internal and external densities of network node

$$\rho_i^{\text{in}} = \frac{\sum_j a_{ij}}{k_i}, \quad i \in h, \quad j \in h, \tag{2}$$

$$\rho_i^{\text{out}} = 1 - \rho_i^{\text{in}}. \tag{3}$$

Here, a_{ij} indicates whether there is a connection between node i and node j. If a connection exists, $a_{ij} = 1$, otherwise $a_{ij} = 0$. The notation h represents the set of communities, while k_i denotes the degree value of node i.

(ii) Calculate the internal and external density entropies of network node

$$H_i = [-\rho_i^{\text{in}} \log(\rho_i^{\text{in}})] + [-\rho_i^{\text{out}} \log(\rho_i^{\text{out}})]. \tag{4}$$

(iii) Calculate the DECS value of network node

$$\text{DECS}(i) = H_i \cdot \frac{k_i}{\sum_{i=1}^{N} k_i}. \tag{5}$$

3.2. Example analysis

In order to enhance the comprehension of the DECS algorithm procedure, the calculation of the DECS value for node 2 is exemplified in the example network as depicted in Fig. 1. In the example network, nodes of different colors are positioned within distinct community structures. Specifically, node 2 shares the same community with its neighbor nodes 1 and 21. The calculation steps for the DECS value of node 2 are as follows.

(i) Calculate the internal and external densities of node 2:

$$\rho_2^{\text{in}} = \frac{\sum_j a_{2j}}{k_2} = \frac{2}{7}, \tag{6}$$

$$\rho_2^{\text{out}} = 1 - \rho_2^{\text{in}} = \frac{5}{7}. \tag{7}$$

(ii) Calculate the internal and external density entropies of node 2:

$$\begin{aligned} H_2 &= [-\rho_2^{\text{in}} \log(\rho_2^{\text{in}})] + [-\rho_2^{\text{out}} \log(\rho_2^{\text{out}})] \\ &= \frac{2}{7} \log \frac{7}{2} + \frac{5}{7} \log \frac{7}{5} = 0.259. \end{aligned} \tag{8}$$

(iii) Calculate the DECS value of node 2:

$$\text{DECS}(2) = H_2 \cdot \frac{k_2}{\sum_{i=1}^{N} k_i} = 0.259 \times \frac{7}{104} = 0.0174. \tag{9}$$

Based on the above calculation steps, the DECS value for node 2 can be computed. It is not possible to differentiate between nodes 8 and 10 in terms of superiority according to the degree centrality algorithm. Nevertheless, node 8 is superior to node 10 based on the DECS values that were computed for both nodes.

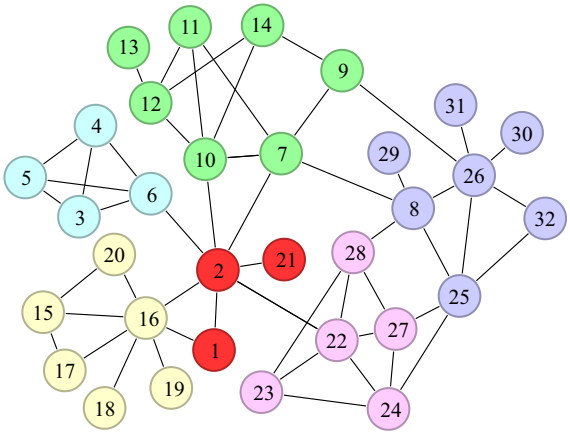


Fig. 1. Example network.

3.3. Time complexity

In this section, this paper will analyze the time complexity of the DECS algorithm, and the specific calculation steps of the algorithm are shown in Algorithm 1. The DECS algorithm is mainly divided into four parts. The first part partitions the network into different communities using the Walktrap algorithm, with a time complexity of $O(MN^2)$. The second part calculates the internal and external densities of network nodes, with a time complexity of $O(NM\langle c\rangle)$. The third part calculates the internal and external density entropies of network nodes, with a time complexity of $O(N)$. The fourth part calculates the DECS value for each node in the network, with a time complexity of $O(1)$. Therefore, the total time complexity of the DECS algorithm is $O(MN^2 + NM\langle c\rangle)$, where N is the number of nodes in the network, M is the number of edges in the network, and $\langle c\rangle$ is the average number of communities in the network. The time complexity of the DECS algorithm is compared with that of the remaining five node centrality algorithms, as shown in Table 1.

Table 1. Complexity analysis of each method.

Centrality method	Time complexity
DC	$O(N^2)\|\|O(M)$
GM	$O(NM+N^2)$
K-shell	$O(N+M)$
LENC	$O(N\langle k\rangle^2)$
SPC	$O(N^2)$
DECS	$O(MN^2+NM\langle c\rangle)$

Algorithm 1 DECS algorithm

Input: Graph $G=(V,E)$, a given network with n nodes and m edges.
Output: the centrality of all nodes based on DECS.
1: By using the Walktrap algorithm, the network can be divided into different communities, denoted as h.
2: **for** $i=1$ to n **do**
3: Calculate the internal and external densities of node i according to Eqs. (2) and (3).
4: Calculate the internal and external density entropies of node i according to Eq. (4).
5: Calculate the DECS value of node i according to Eq. (5).
6: **end for**
7: **return** the centrality of all nodes based on DECS.

4. Experimental setup

4.1. Datasets

In order to verify the performance of the algorithm, this paper selects eight different-sized network datasets, all of which are unweighted undirected graphs. The eight networks are as follows:

a) Club,[31] the karate club network reflects the social relationships among members of a karate club at an American university. Nodes represent club members, and edges represent social relationships.

b) Adjnoun,[32] the Adjnoun network is made up of 112 common adjectives and nouns that are connected to one another via neighbor phrases in the novel David Copperfield written by Charles Dickens. Nodes represent adjectives or nouns, and edges indicate their co-occurrence in any adjacent positions.

c) Trapping,[33] a real-world animal interaction network. The animal interaction data comes from studies on wild and domestic animals. Nodes represent animals, and edges represent captured interactions.

d) CE-HT,[34] a biological network of gene functional correlations. Nodes represent genes, and edges represent functional relationships between genes.

e) Openflights,[35] the infrastructure network of flights between world airports. Nodes represent airports, and edges represent flight routes.

f) PowerGrid,[36] the network is created based on the topological structure of the Western United States power grid. Nodes represent substations, and edges represent transmission lines.

g) BA,[37] this is a synthetic scale-free network with 1000 nodes, and 2 edges added each time, and it contains 19996 edges.

h) Ca-HepPH,[38] the Ca-HepPh network is a collaborative network where the nodes are authors of articles that have been published, and the edges indicate that two or more authors have co-authored papers.

The statistical characteristics of the networks are shown in Table 2, where N represents the number of nodes in the network, E represents the number of edges between network nodes, $\langle k\rangle$ represents the average degree of the network, $K_{\max}$ represents the maximum degree of network nodes, C represents the average clustering coefficient of the network, and $\langle d\rangle$ represents the average shortest path length in the network.

Table 2. Statistical characteristics of the network.

Network	N	E	$\langle k\rangle$	$K_{\max}$	C	$\langle d\rangle$
Club	34	78	4.588	17	0.588	2.408
Adjnoun	112	425	7.589	49	0.190	2.536
Trapping	1686	4623	5.484	37	0.537	6.536
CE-HT	2617	2985	2.281	44	0.018	8.644
Openflights	2939	15677	10.668	242	0.589	4.097
PowerGrid	4941	6594	2.669	19	0.107	18.989
BA	10000	19996	3.999	334	0.005	4.996
Ca-HepPH	11204	117619	20.996	491	0.690	4.673

4.2. Reference algorithms

In this paper, we conducted experimental comparisons of five node centrality algorithms, including classic algorithms such as degree centrality (DC) and K-shell. We also selected the gravity model-based centrality method (GM), entropy-based centrality method (LENC) and node spreading probability-based centrality method (SPC).

(i) **DC**[39] Degree centrality is a simple method of ranking node centrality. The more neighbor nodes a target node has, the greater the degree value and importance of the node.

(ii) **GM**[13] The gravity model is a method that considers neighborhood and path information to identify influential nodes. It believes the node with more neighbors and lower average distances to be more influential.

(iii) **K-shell**[40] The position of a node in a network plays a crucial role in measuring its impact. The higher the K-shell value, the more central the node is in the network, indicating its greater importance.

(iv) **LENC**[41] It is an entropy-based centrality algorithm, taking into account the entropy and the weight distribution of the edges.

(v) **SPC**[42] The centrality algorithm is based on node spreading probability, measuring the propagation ability of each node in the network.

4.3. Evaluation metrics

In each network, this paper selects the top-1 node under different centrality algorithms as the infection source for conducting SIR propagation experiments. Meanwhile, propagation influence and AUC value are utilized as the evaluation metrics for measuring the performance of node centrality algorithms, where the propagation influence $F(t)$ represents the proportion of the sum of infected and recovered nodes to the total number of nodes.

Propagation influence Given an undirected graph $G = (V,E)$ with N nodes, the propagation influence $F(t)$ at time t under the SIR propagation model can be defined as

$$F(t) = \frac{I+R}{N}, \tag{10}$$

where I is the number of infected nodes, R is the number of recovered nodes, and N is the total number of nodes in the network.

AUC (area under the curve) In order to quantify the infection capability of the top-1 node under different centrality algorithms, this paper uses AUC to represent the performance of the node centrality algorithm. AUC is a performance metric for classification models, commonly used to evaluate the predictive ability of binary classification models at different thresholds.[43] The AUC value ranges from 0 to 1, with a higher value indicating better model performance. The value of the AUC can be calculated by plotting the ROC curve (receiver operating characteristic curve), where the AUC value is equal to the area under the ROC curve. The area enclosed by the ROC curve and the x-axis is transformed into a bar graph, representing the normalization of AUC. The normalized AUC value is defined as follows:

$$\mathrm{AUC} = \frac{S_i}{\max\{S_1, S_2, \ldots, S_6\}}, \quad i = 1,2,\ldots,6, \tag{11}$$

where S_i represents the area enclosed by the i-th ROC curve on the x-axis.

4.4. Result and analysis

4.4.1. SIR propagation experiments

For the aforementioned eight networks, the first node under different centrality methods is selected as the initial infected node, while the remaining nodes are considered susceptible nodes. The top-1 node is chosen as the infection source for conducting SIR propagation experiments. In Fig. 2, the x-axis represents epoch as the time step, and the y-axis $F(t)$ represents the proportion of the number of infected and recovered nodes to the total number of nodes. The brown curve in Fig. 2 represents the experimental results obtained using the DECS algorithm in this paper. In the Club network, the experimental results reveal that the initial-stage performance of node propagation using the DECS algorithm surpasses that of other centrality methods. However, after over 4 iterations, the performance of the DECS algorithm slightly lags behind that of SPC and K-shell. As for the Adjnoun network, minimal disparity in performance is observed between the DECS algorithm and other comparison algorithms in the early stage of the experiment.

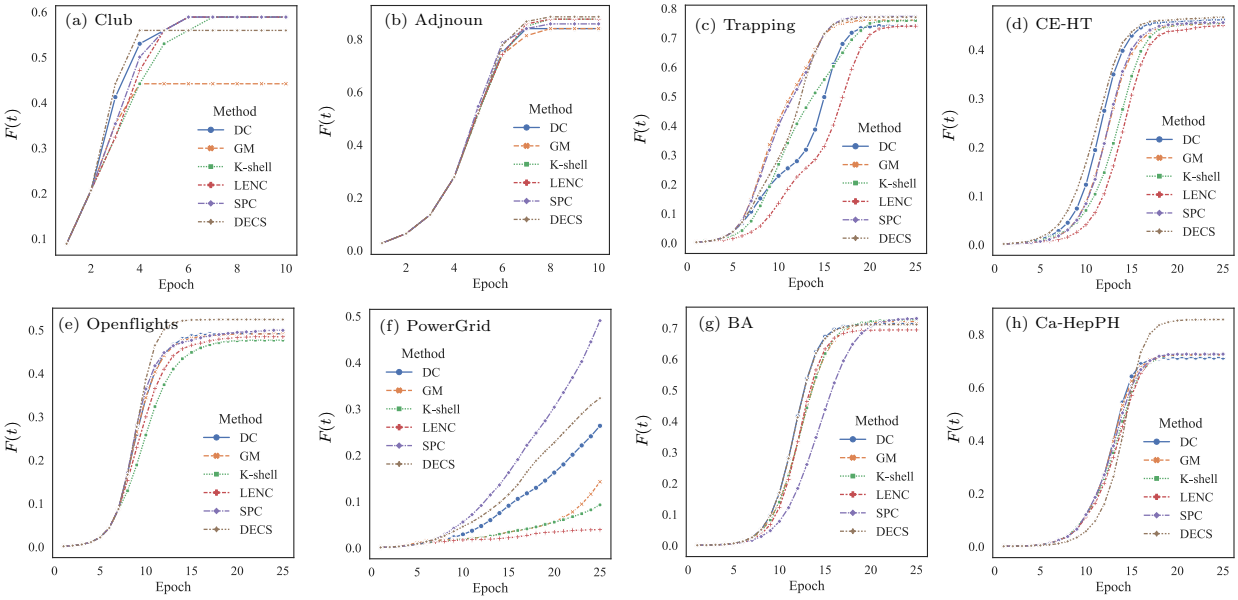


Fig. 2. Infection ratio of top-1 nodes in eight networks under different methods: (a) Club; (b) Adjnoun; (c) Trapping; (d) CE-HT; (e) Openflights; (f) PowerGrid; (g) BA; (h) Ca-HepPH.

Surpassing the performance of the other five centrality methods occurs when the number of iterations exceeds 6. In the Trapping network, the initial propagation effect is marginally inferior to that of GM and SPC. Nevertheless, after more than 14 iterations, the DECS algorithm can achieve the same propagation effect as GM and SPC. In the CE-HT network, the DECS algorithm demonstrates a markedly superior propagation effect compared to other node centrality methods. In the Openflights network, the DECS algorithm holds a distinct advantage when the number of iterations exceeds 9. In the PowerGrid network, the propagation effect of the DECS algorithm is slightly lower than that of SPC. In the BA scale-free network, the propagation effect of the DECS algorithm is second only to DC and is significantly better than SPC. In the Ca-HepPH network, the early-stage propagation effects of the DECS algorithm are moderate, but when the number of iterations exceeds 16, the DECS algorithm is significantly superior to the other five algorithms.

As shown in Fig. 3, the AUC results of the node infection ratio curves generated by different centrality methods in Fig. 2 are presented. A higher AUC value indicates better effectiveness of the centrality algorithm for the nodes, and the DECS algorithm is represented by the brown bar chart. From Fig. 3, it can be observed that the DECS algorithm performs well in the Ca-HepPH, Openflights, CE-HT, and Adjnoun networks. In the Club network, the DECS algorithm outperforms GM. In the PowerGrid and Trapping networks, the DECS algorithm ranks second, only surpassed by SPC. In the BA network, the DECS algorithm outperforms both LENC and SPC. Overall, the DECS algorithm is capable of effectively identifying nodes with strong propagation abilities.

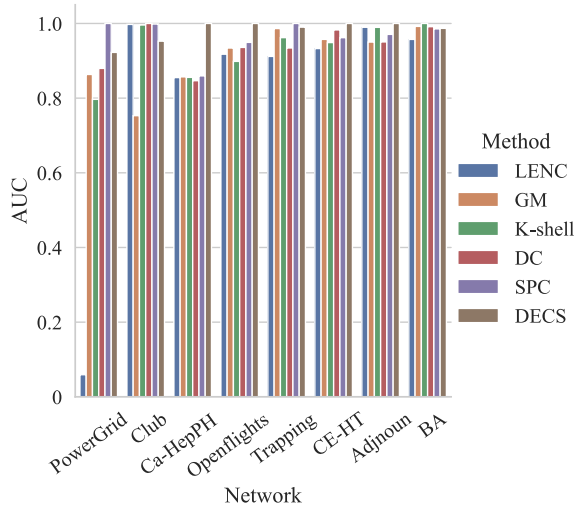


Fig. 3. AUC results of the infection ratio of top-1 nodes under different methods in eight networks.

4.4.2. Network static attack experiments

Static attack experiments are conducted to evaluate the accuracy of node centrality methods based on the stability of the network. When nodes and their corresponding edges are removed from the network, the structure of the network will change. The greater the degree of network structure disruption, the more critical the removed nodes are.[44,45] In this paper, the network status is observed through static attack experiments. The specific process is as follows.

(i) Calculate the influence of different centrality methods on the nodes and order the network nodes in descending order based on the centrality results.

(ii) Remove the top-ranked important nodes and their corresponding edges from the network.

(iii) Observe and record the network's status after each node removal.

(iv) Repeat steps 2 and 3 until 30% of the nodes are removed from the network.

To measure the changes in the network structure, the size of the giant component and the number of components are used as indicators. The size of the giant component reflects the connectivity of the network's structure. At the same time, the number of components also indicates the impact of node removal on network fragmentation. The fewer nodes in the giant component, the poorer the network's connectivity, and the more important the removed nodes are in the network.

As shown in Fig. 4, the changes in the giant component size of eight networks are shown in static attack experiments. In the Club network, when the proportion of removed nodes reaches 18%, the trend of change in the giant component for the DECS algorithm approaches that of DC. When the removed nodes reach 24%, the DECS algorithm surpasses other centrality methods. In the Adjnoun, Trapping, and CE-HT networks, the rapid and superior changes in the giant component size outperform those achieved by the K-shell, LENC, and SPC algorithms. In the Openflights and PowerGrid networks, the giant component size experiences a rapid reduction as the deleted nodes reach 12%, with the DECS algorithm consistently exhibiting superior performance compared to other centrality methods. In the BA network, a significant decrease in the number of the giant component is observed when the number of removed nodes reaches 18%, with the DECS algorithm only surpassed by DC. In the Ca-HepPH network, the DECS algorithm demonstrates moderate attack effectiveness and underperforms in densely connected large-scale networks.

Figure 5 shows the changes in the number of components in the eight networks in the static attack experiments. In the Club network, the DECS algorithm produces results similar to DC when the number of deleted nodes is below 18%. However, its performance surpasses other centrality methods when more than 24% of nodes are deleted. In the Adjnoun, Trapping, and CE-HT networks, the DECS algorithm performs better than all centrality methods except for DC. In the Openflights network, the DECS algorithm performs better than other centrality methods when the number of deleted

nodes is less than 12%, but there is a slight decrease in the number of components when it exceeds 12%. However, it still outperforms the K-shell, LENC, and SPC algorithms. In the PowerGrid network, the DECS algorithm is second only to DC when the number of deleted nodes is less than 20%, but there is a decrease in the number of components when it exceeds 20%. However, it still outperforms the K-shell algorithm. In the Ca-HepPH network, the DECS algorithm demonstrates only moderate performance. In the BA network, the DECS algorithm exhibits similar performance to DC and outperforms the other four node centrality methods. Overall, the DECS algorithm can effectively and accurately evaluate the centrality of nodes.

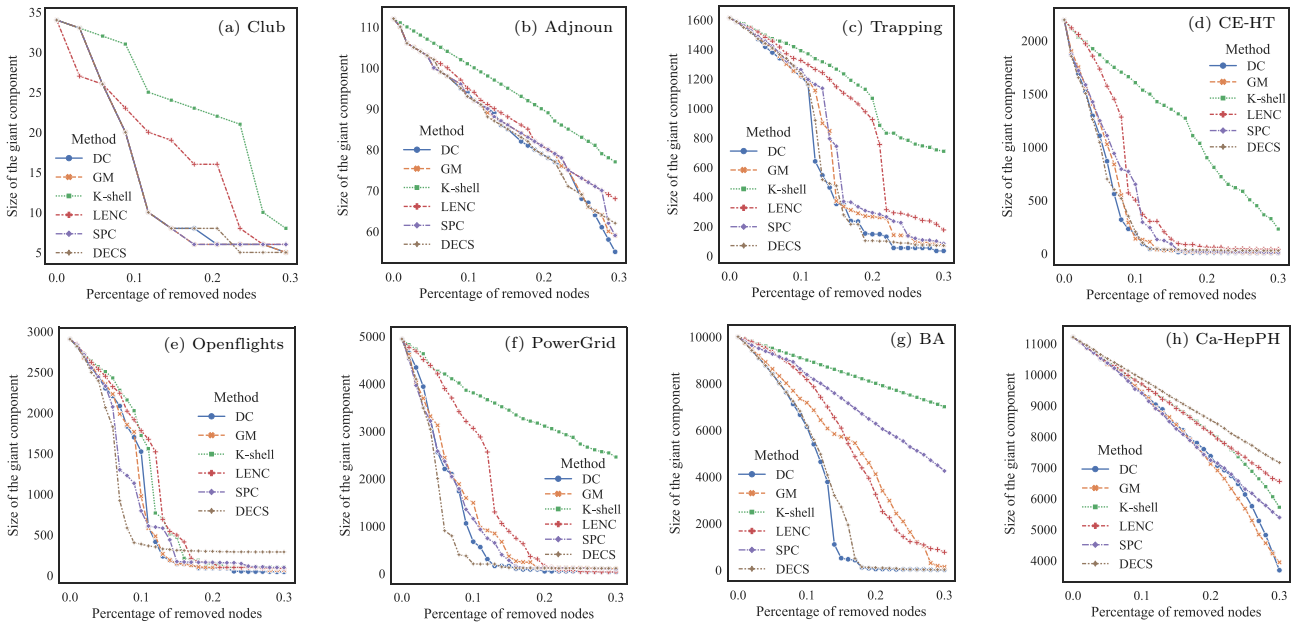


Fig. 4. Size of the giant component in eight networks during static attack experiments: (a) Club; (b) Adjnoun; (c) Trapping; (d) CE-HT; (e) Openflights; (f) PowerGrid; (g) BA; (h) Ca-HepPH.

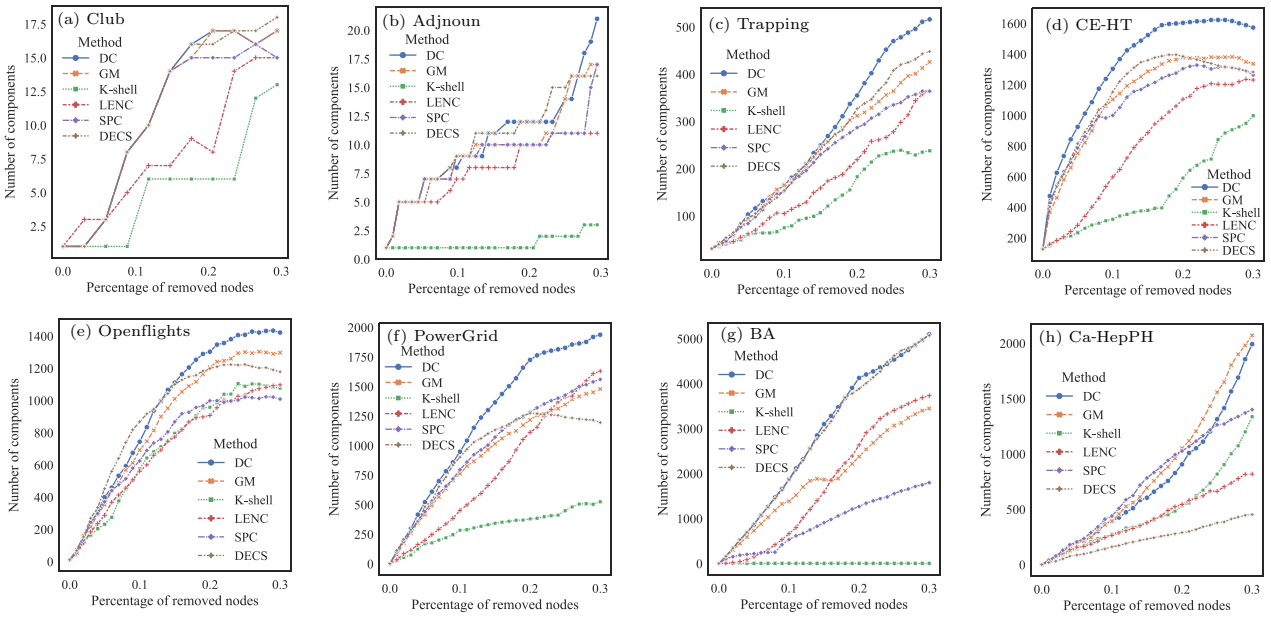


Fig. 5. Number of components in eight networks during static attack experiments: (a) Club; (b) Adjnoun; (c) Trapping; (d) CE-HT; (e) Openflights; (f) PowerGrid; (g) BA; (h) Ca-HepPH.

In order to measure the performance of node centrality algorithms, this paper uses normalized AUC results to evaluate the effects of different centrality algorithms in static attack experiments. As shown in Fig. 6, it presents the normalized AUC results of the size of the giant component for different centrality methods in static attack experiments from Fig. 4, where a smaller AUC value indicates a better attack effect for the node centrality algorithm, and the DECS algorithm is represented by the blue bars. In the CE-HT, BA, and Trapping networks, the DECS algorithm performs second only to DC and outperforms the other four centrality methods. In the PowerGrid and Openflights networks, the DECS algorithm outperforms

the other five centrality methods. In the Ca-HepPH network, the performance of the DECS algorithm is moderate due to the large and dense nature of the network. In the Club and Adjnoun networks, the DECS algorithm performs similarly to DC, demonstrating superiority over the K-shell and LENC algorithms.

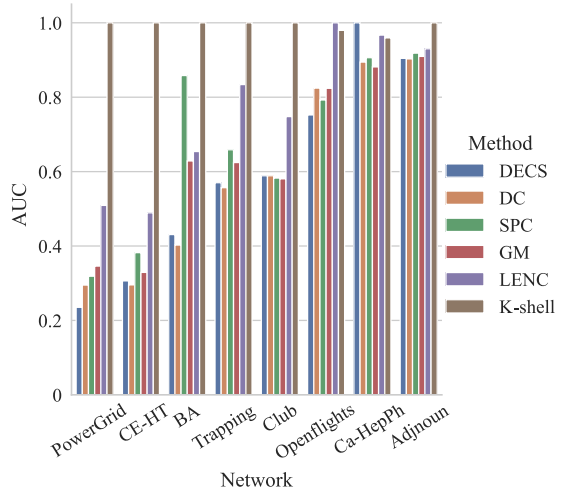


Fig. 6. AUC results of the size of the giant component in eight networks during static attack experiments.

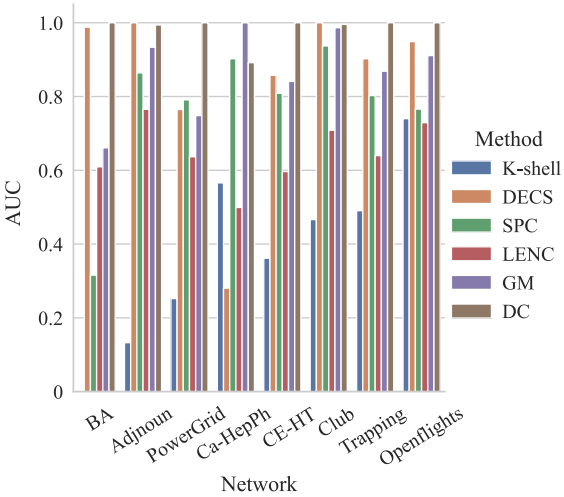


Fig. 7. AUC results of the number of components in eight networks during static attack experiments.

As shown in Fig. 7, it shows the normalized AUC results of the number of components for different centrality methods in the static attack experiments from Fig. 5, where a larger AUC value indicates a better attack effect for the node centrality algorithm, and the DECS algorithm is represented by the orange bars. In the CE-HT, BA, Trapping, and Openflights networks, the DECS algorithm ranks second only to DC. In the PowerGrid network, the DECS algorithm outperforms the K-shell, LENC, and GM algorithms. However, its performance in the Ca-HepPH network is hindered by the extensive scale and dense interconnectivity among network nodes. In the Club and Adjnoun networks, the DECS algorithm surpasses all five centrality methods. These findings collectively demonstrate the DECS algorithm's effectiveness in identifying key nodes in networks, thereby enhancing the accuracy of evaluating node centrality methods.

5. Conclusion

To take into account the impact of community structure on the centrality of target nodes in a network and further identify highly influential nodes with strong propagation capabilities, this paper proposes an influential spreaders identification algorithm based on density entropy and community structure (DECS). In this paper, the Walktrap community detection algorithm is used to partition the network into communities. Additionally, the centrality of the target nodes is measured by considering their internal and external density entropies as well as their degree values.

SIR propagation experiments and static attack experiments were conducted on eight networks. The experimental results show that nodes with higher DECS values are more likely to act as infection sources, resulting in a larger scale of network infection. In the static attack experiments, the DECS method demonstrates good attack effectiveness in the majority of cases, accurately assessing the influence of nodes. The DECS algorithm can effectively identify key nodes in a network, which serve as bridges connecting different communities. At the same time, this algorithm comprehensively considers the community structure and node degrees of the network, thus better uncovering important nodes with stronger propagation capabilities in the network. The proposed DECS algorithm is more suitable for medium-sized networks with a high average degree, as networks with a higher average degree have more neighbor nodes for the target node, allowing for better utilization of the node's internal and external density entropies to calculate its influence.

However, the community structure of networks, obtained through the Walktrap algorithm, is not perfect, and there is still room for further improvement and optimization in the community partition process. In addition, the performance of the DECS algorithm is suboptimal in densely connected large-scale networks and requires further improvement in future research. Due to the limitations of this paper, the DECS algorithm did not take into consideration weighted networks. The next stage of our work will focus on integrating weighted degree centrality with network community structure to calculate node centrality in weighted networks. At the same time, the centrality of nodes in directed networks will also be a primary focus of our future research.

Acknowledgement

Project supported by the National Natural Science Foundation of China (Grant No. 61803264).

References

[1] Liu Y, Song A, Shan X, Xue Y and Jin J 2022 *Expert Syst. Appl.* **196** 116557
[2] Munikoti S, Das L and Natarajan B 2022 *Neurocomputing* **468** 211
[3] Tikka V, Haapaniemi J, Räisänen O and Honkapuro S 2022 *Appl. Energy* **328** 120124
[4] Zhang H, Fiszman M, Shin D, Miller C M, Rosemblat G and Rindflesch T C 2011 *J. Biomed. Inform.* **44** 830
[5] Liu Y, Jiang M, Hu L and He Z 2023 *J. Informetr.* **17** 101424
[6] Freeman L C 1977 *Sociometry* **40** 35
[7] Brandes U, Borgatti S P and Freeman L C 2016 *Soc. Netw.* **44** 153
[8] Xu Q, Sun L Z and Bu C J 2023 *Chaos Solitons Fractals* **173** 113753
[9] Zhang Q, Tang R, Yao Z and Zhang Z B 2023 *Appl. Math. Comput.* **459** 128276
[10] Qiu L, Zhang J and Tian X 2021 *Appl. Intell.* **51** 4394
[11] Asgharian Rezaei A, Munoz J, Jalili M and Khayyam H 2023 *Expert Syst. Appl.* **214** 119086
[12] Zhao G, Jia P, Zhou A and Zhang B 2020 *Neurocomputing* **414** 18
[13] Li Z, Ren T, Ma X, Liu S, Zhang Y and Zhou T 2019 *Sci. Rep.* **9** 8387
[14] Yang X and Xiao F 2021 *Knowl-Based Syst.* **227** 107198
[15] Li Z and Huang X 2022 *Sci. Rep.* **12** 9879
[16] Zhao Z, Li D, Sun Y, Zhang R and Liu J 2023 *Knowl-Based Syst.* **260** 110163
[17] Han Z M, Wu Y, Tan X S, Duan D G and Yang W J 2015 *Acta Phys. Sin.* **64** 058902 (in Chinese)
[18] Jiang Y, Yang S Q, Yan Y W, Tong T C and Dai J Y 2022 *Chin. Phys. B* **31** 058903
[19] Han Z M, Chen Y, Li M Q, Liu W and Yang W J 2016 *Acta Phys. Sin.* **65** 168901 (in Chinese)
[20] Ma X J and Ma Y H 2019 *Complexity* **2019** 9057194
[21] Zhong L F, Bai Y, Tian Y, Luo C, Huang J and Pan W J 2021 *Complexity* **2021** 5554322
[22] Wang T T, Liang Z W and Zhang R X 2023 *Acta Phys. Sin.* **72** 048901 (in Chinese)
[23] Kumar S, Kumar A and Panda B S 2023 *IEEE Trans. Ind. Inf.* **19** 703
[24] Sun P G, Miao Q and Staab S 2021 *Pattern Recognition* **120** 108130
[25] Zhao Z J, Guo Q, Yu K and Liu J G 2020 *Physica A* **551** 123893
[26] Pons P and Latapy M 2006 *J. Graph Algorithms Appl.* **10** 191
[27] Hu F and Liu Y 2016 *Physica A* **446** 138
[28] Saraiva P 2023 *Kuwait J. Sci.* **50** 194
[29] Li X, Zhang X, Zhao C and Duan X 2021 *Chaos* **31** 051104
[30] Zhu L, Yang F, Guan G and Zhang Z 2021 *Inf. Sci.* **562** 240
[31] Gupta M and Mishra R 2021 *Decis. Support Syst.* **149** 113608
[32] Yang X H, Xiong Z, Ma F, Chen X, Ruan Z, Jiang P and Xu X 2021 *Physica A* **573** 125971
[33] Rossi R and Ahmed N 2015 *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, January 25–30, 2015, Austin, Texas, USA, p. 4292
[34] Yang K, Liu Y, Zhao Z, Zhou X, Ding P 2023 *Eur. Phys. J. B* **96** 27
[35] Liu C, Wang P and Chen A *2021* 40*th Chinese Control Conference (CCC)*, July 26–28, 2021, Shanghai, China, p. 736
[36] Watts D J and Strogatz S H 1998 *Nature* **393** 440
[37] Barabási A L and Albert R 1999 *Science* **286** 509
[38] Leskovec J, Kleinberg J and Faloutsos C 2007 *ACM Trans. Knowl. Discov. Data* **1** 2
[39] Freeman L C 1978 *Soc. Netw.* **1** 215
[40] Kitsak M, Gallos L K, Havlin S, Liljeros F, Muchnik L, Stanley H E and Makse H A 2010 *Nat. Phys.* **6** 888
[41] Wang B, Zhang J, Dai J and Sheng J 2022 *Sci. Rep.* **12** 1833
[42] Ai J, He T, Su Z and Shang L H 2022 *Chaos Solitons Fractals* **164** 112627
[43] Kumar S, Mallik A and Panda B S 2022 *World Wide Web* **25** 2487
[44] Yuan Z W, Lv C C, Si S B and Duan D L 2021 *Chin. Phys. B* **30** 050501
[45] Zhou F, Yuan Y and Zhang M 2019 *Arab. J. Sci. Eng.* **44** 2837