Available online at sjuoz.uoz.edu.krd

Science Journal of University of Zakho
Vol. 13, No.2 pp235–244April-Jun,2025

p-ISSN: 2663-628X
e-ISSN: 2663-6298

LCD: IDENTIFYING INFLUENTIAL NODES IN COMPLEX NETWORKS WITH LAYERED CLUSTERING AND DEGREE

Abdulhakeem O. Mohammed [1,*]

[1]Department of Computer Science, College of Science, University of Zakho, Zakho, Kurdistan Region, Iraq.

*Corresponding author email: a.mohammed@uoz.edu.krd

Received: 13 Nov., 2024 Accepted:24 Mar., 2025 / Published: 14 Apr., 2025. https://doi.org/10.25271/sjuoz.2025.13.2.1483

ABSTRACT:

Identifying influential nodes in networks is a key challenge in understanding how information spreads. While numerous algorithms have been proposed in the literature, many struggle with either limited spreading efficiency or high computational complexity. To address this challenge, we present Layered Clustering Degree (LCD), an effective method for identifying a set of well-distributed spreaders with high spreading ability, while maintaining a computational complexity of $O(V + E)$, making it highly suitable for large-scale networks where both efficiency and computational complexity are essential. The LCD approach operates in three main steps: (1) Layering, which organizes nodes hierarchically based on their shortest distances from a designated starting node; (2) Clustering, which groups nodes within each layer into connected substructures to capture local connectivity patterns; and (3) Degree computation and ranking, where node degrees are computed within the entire network (globally) but ranked iteratively across clusters (locally) to ensure maximum coverage and minimal overlap among selected spreaders. The significance of layered clustering in LCD method, is iteratively distributing spreaders across clusters to avoid over-representation of high-degree nodes from a single region of the network. Experimental results using the SIR model on nine real-world networks show that LCD outperforms several popular methods, including VoteRank, K-shell, VoteRank++, ClusterRank, H-Index, EnRenew, and DegreeRank, in terms of spreading rate and final affected scale.

KEYWORDS: Complex Networks; Influential Nodes; SIR Model; Layered Clustering; Degree

1. INTRODUCTION

Identification of influential nodes in complex networks is a very challenging task with extensive applications in numerous fields. In social networks, the recognition of influential individuals can greatly facilitate information diffusion (Lu *et al*., 2011; Chen *et al*., 2014), more efficient conduits for the dissemination of ideas (Centola, 2010), trends, or advertising campaigns (Sheikhahmadi and Nematbakhsh, 2017). In epidemiology, the identification of crucial nodes in disease networks is of paramount importance, as it provide insights for controlling and preventing the spread of infectious diseases (Pastor-Satorras *et al*., 2015; Ouboter *et al*., 2016; Buscarino *et al*., 2008; Pastor-Satorras and Vespignani, 2001). Thus, this problem is a foundation for network optimization, information propagation, and public health strategies.

A wide range of algorithms has been developed in the literature to identify influential nodes in complex networks, each offering unique approaches to the challenge. Some algorithms consider node's local information, such as degree centrality (Freeman *et al*., 2002) which counts the number of node's direct neighbors as its influence. H-index (Lu *et al*., 2016) considers second order neighbors as node's significance. Based on degree centrality, LocalRank (Chen *et al*., 2012) takes into account node's 4-th order neighbors. Furthermore, Liu *et al*. proposed a neighborhood centrality measure that integrates a node's centrality and its neighbors' centralities in two steps (Liu *et al*., 2016). They showed that a 2-step neighborhood provides the best balance between computational cost and ranking performance, and extending beyond that provides limited or negative improvements in accuracy. ClusterRank (Chen *et al*., 2013) combines clustering coefficient and Degree centrality to measure node's importance in spreading. Kitsak *et al*. introduced the k-shell decomposition method, which ranks influential nodes based on their structural position, rather than their direct connections (Kitsak *et al*., 2010). This method decomposes the network into hierarchical layers in which nodes with higher k-shell values occupy the core position of the network. The method suggests that the centrality of a node relies primarily on its position in the network structure, with the most central nodes being those in the innermost shells. The k-shell method, however, does not give a unique ranking because multiple nodes may share the same k-shell values. To avoid this drawback, Bae and Kim (Bae and Kim, 2014) suggested using coreness centrality, which takes into consideration the k-shell values of adjacent nodes. Zeng and Zhang presented algorithm for ranking nodes by decomposing networks in a mixed degree decomposition process (Zeng and Zhang, 2013). Chen *et al*. proposed the DegreeDiscount algorithm (Chen *et al*., 2009), which simplifies the process by reducing the degree of a chosen node's neighbors.

Local metrics, while straightforward, tend to be less effective due to their neglect of the broader network structure. Common global metrics, such as closeness centrality (Sabidussi, 1966), betweenness centrality (Freeman, 1977), and Katz centrality (Katz, 1953), excel in identifying influential nodes but are associated with high computational costs. As a result, they are often considered impractical for large-scale networks.

Recently, many other algorithms based on various heuristics and ideas have been proposed (Guo *et al*., 2020; Liu *et al*., 2021; Wang *et al*., 2022; Zhao et *al*., 2023; Xu and Dong, 2024; Yang *et al*., 2024). Zhang *et al*. introduced a method called VoteRank, aimed at identifying influential spreaders in complex networks

* Corresponding author

(Zhang *et al*., 2016). This method involves node selection techniques designed to minimize overlapping influence at both individual and group levels. Kumar and Panda developed this approach further with the Neighborhood Coreness approach, which incorporates the k-shell values of the neighbors to increase the resolution of VoteRank (Kumar and Panda, 2020). Despite improvements in these methods, overlapping influence areas between spreaders has remained a persistent issue. Liu *et al*. proposed the new concept of VoteRank++ to improve the detection of influential nodes in complex networks (Liu et al., 2021). VoteRank++ assigns voting weights based on whether the nodes have the same degree or different degrees, and the degree of closeness between these nodes is determined by the influence that a node can have on its neighbors. Guo *et al*. Introduced the EnRenew algorithm, which improves the node selection based on information entropy (Guo *et al*., 2020). Given the initial spreading ability as the entropy of each node, the algorithm selects the nodes whose entropy is maximum and iteratively updates the spreading capability of all reachable nodes by reducing their influence using an attenuation factor until the desired expected number of influential nodes is selected.

Despite the advantages of current methods, an essential challenge remains in balancing the effectiveness of information spreading with computational efficiency. Additionally, current methods generally exhibit a critical limitation in the potential redundancy of influence ranges when the identified spreaders are close to each other. This overlap of influence areas reduces the overall influence efficiency of chosen influential nodes, since their influence areas intersect instead of reaching out to unexplored regions of the network. This proximity relation among highly ranked nodes can drastically decrease the overall coverage and performance of information dissemination.

To address these challenges, we propose the Layered Clustering Degree (LCD) method, a simple yet effective approach to influential node identification. LCD works in three major steps. Firstly, layering process positions nodes in a hierarchy according to their minimum distances from any given starting node, usually the one with the highest eccentricity from any random starting node. This step splits the network into layers, providing a structured framework for analyzing connectivity. Then, in the clustering phase, it groups nodes within each layer into connected substructures to reveal local connectivity patterns and to ensure balanced influence distribution. Finally, in the degree computation and ranking step, node degrees are computed globally in the whole network, but ranking is performed iteratively within clusters. This new approach prevents the over-representation of high-degree nodes from one cluster and allows the spreading of influential nodes across the network. Iterative ranking is important, with maximum coverage and minimum redundancy, leading to effective influence propagation. We perform comprehensive experiments on nine real-world networks and show that LCD outperforms state-of-the-art methods, such as VoteRank, K-shell, VoteRank++, ClusterRank, H-Index, EnRenew, and DegreeRank, in terms spreading rate and final influenced scale. Moreover, our method addresses the computational efficiency limitations of existing algorithms. While methods like *VoteRank++* may achieve good spreading results but suffer from high computational costs, *LCD* maintains better spreading performance with a more efficient time complexity of $O(V + E)$, making it practical for large-scale network analysis. Additionally, experimental results show that *LCD* significantly outperforms traditional degree centrality, with the top-k influential nodes identified by *LCD* achieving substantially wider and faster spreading. The source code for LCD, along with documentation and experimental results, is publicly available at GitHub.

2. PROPOSED METHOD AND MATERIALS

Preliminaries

Consider a connected, undirected network $G = (V, E)$, where V and E denote the set of nodes and edges in the network respectively, n and m to will be used Interchangeably to denote the number of nodes and edges, respectively. The distance between nodes u and v , denoted as $d_G\ (u, v)$, is the number of edges in the shortest path connecting them. The eccentricity of a node v, denoted by $ecc(v)$, is the largest distance from v to any other node, i.e., $ecc(v) = max_{u \in V} d_G(u, v)$. For a node v of G, $N(v) = \{u \in V : uv \in E\}$ is called the neighborhood of v.

The concept of layered clustering, introduced by Chepoi and Dragan (Chepoi and Dragan, 2000), provides a framework for partitioning the nodes based on both distance and connectivity patterns. Given a starting node s, the nodes of G are first divided into distance layers, and each layer is further partitioned into clusters. Two vertices u and v within the same layer are grouped into the same cluster if they are connected by a path that uses only nodes within the same or higher layers. More formally, a layering of a graph $G = (V, E)$ with respect to a start node s is the decomposition of V into $r + 1$ layers, where $r = ecc(s)$

$$L_i\ (s) = \{u \in V : d_G\ (s, u) = i\},\ \ i = 0,1, \dots, r$$

A layered clustering of G, denoted as

$$LC(G, s) = \{L_i^1, \dots, L_i^{p_i} : i = 0, 1, \dots, r\}$$

partitions each layer $L_i\ (s)$ into clusters $L_i^1\ , \dots, L_i^{pi}$ such that two nodes $u, v \in L_i\ (s)$ belong to the same cluster L_i^j if and only if they can be connected by a path outside the ball $B_{i-1}\ (s)$ of radius $i - 1$ centered at s. Here, pi is the number of clusters in layer i. It was shown in (Chepoi and Dragan, 2000) that for a given graph G, a layered clustering can be found in $O(n + m)$ time. See Section 3.1 for an example and further insights into layered clustering.

Spreading Model

The Susceptible-Infected-Recovered (*SIR*) model (Tao *et al*., 2006) is a widely used epidemic spreading model for simulating and analyzing propagation dynamics in networks, such as disease outbreaks (Anderson and May, 1991; Hethcote, 2000; Buscarino *et al*., 2008), information diffusion (Zhao *et al*., 2013), and rumor spreading (Zhao *et al*., 2013). Recently, this model has been frequently used for evaluating the performance of influential nodes identification algorithms in complex networks (Kitsak *et al*., 2010; Zhang *et al*., 2016; Guo *et al*., 2020; Liu *et al*., 2021; Yang and Xiao, 2021). Each node in the network can exist in one of three states: Susceptible (*S*), capable of receiving the information or infection; Infected (*I*), actively spreading the information or disease; and Recovered (*R*), which no longer participates in the spread. There are various adaptations of the SIR model that differ in how infected nodes interact with their neighbors, such as the limited contact variation (Zhang *et al*., 2016; Guo *et al*., 2020) and the full contact variation (Wang *et al*., 2023; Liu *et al*., 2021). In this study, we adopt the limited contact variation.

Initially, a selected set of nodes (seed nodes) are marked as infected, while all other nodes are susceptible. During each time step t, an infected node transmits the information or infection to its susceptible neighbors with a probability μ (infection probability) and transitions to the recovered state with a probability β (recovery probability). The process terminates when no infected nodes remain in the network, indicating a steady state. The infected rate $\lambda = \mu/\beta$ plays a critical role in determining the spread's speed and scale. To ensure meaningful

comparisons in experiments, the infection probability is typically set to $\mu = 1.5\mu_c$, where $\mu_c = (< k >)/(< k^2 > - < k >)$ is the spreading threshold calculated based on the network's structural properties (Castellano and Pastor-Satorras, 2010), $< k >$ and $< k^2 >$ denote the average degree and the second order average degree, respectively. Simulations are run multiple times to account for randomness, and the results are averaged for robust evaluation of spreading efficiency. The *SIR* model's simplicity and effectiveness make it a popular tool for evaluating the influence of nodes and the performance of influential nodes identification algorithms in complex networks. Metrics based on the spreading scale under the *SIR* model are detailed in Section 2.2.

LCD Algorithm

In this section, the Layered Clustering Degree (LCD) algorithm is introduced as an efficient method for identifying influential nodes in complex networks. The LCD algorithm is inspired by the layered clustering principle, as introduced in (Chepoi and Dragan, 2000), that emphasizes hierarchical network analysis by partitioning graphs into layers and clusters reflecting the core structure of the network (see Section 2.1 for more details). Building on this concept, the LCD algorithm combines hierarchical layering, clustering, and degree-based ranking to achieve balanced and effective node influence identification. The LCD algorithm tackles the drawbacks of traditional algorithms, such as redundancy in selected spreaders and lack of coverage across the network, in a systematic manner while being computationally efficient. LCD not only finds globally influential nodes but also spreads these nodes over various parts of the network. This balanced approach improves the efficiency of spreading processes such information diffusion, epidemic control, and resource optimization.

The algorithm starts with the layering step by dividing the network into hierarchical layers based on their shortest distances from a designated starting nod. This starting node is typically chosen as the one with the maximum eccentricity relative to a randomly selected initial node. Then the clustering step groups nodes within each layer into connected substructures to capture local connectivity patterns and guarantee balanced influence distribution. Lastly, in the degree calculation and ranking step, node degrees are calculated globally from the whole network, but ranking is done iteratively on a per-cluster basis. By computing degrees globally, the approach prevents underestimating a node's influence due to clustering localized effects. This iterative process of ranking round by round guarantees that influential nodes are spread out across the network while their context is preserved in their respective layers and clusters. By prioritizing the highest degree nodes in each cluster, the *LCD* algorithm mitigates the issue of multiple high-degree nodes from the same cluster (from the same localized area in the network) dominating the rankings, ensuring broader coverage. For further clarity, an illustrative example can be found in Section 3.1.

A detailed step-by-step explanation of the LCD algorithm is presented below.

1. Layering: The LCD algorithm begins by organizing the network into hierarchical layers. To do that, randomly select an initial node w from the graph and then a starting node s with the maximum eccentricity from w is identified to ensure that the layering process captures the global structure of the network.
   - Using Breadth-First Search (*BFS*) algorithm from s, the shortest paths from this node to all other nodes are computed.
   - Nodes are then grouped into layers based on their distance from the starting node s. The i^{th} layer contains nodes whose distance from the s is exactly i.
2. Clustering: Once the layers are established, the algorithm creates clusters within each layer to group nodes with strong local connectivity. Divide each layer into clusters such that two nodes u and v belong to the same cluster if they are connected by a path using only nodes in the same or higher layers.
   - For each layer, a subgraph is constructed consisting of the nodes in the layer and the edges between them.
   - BFS traversal is performed within each subgraph to identify connected components, which serve as the clusters for the layer.
3. Degree computation and ranking: Within each cluster, compute the degrees of its nodes across the entire network (globally), then sort the nodes in descending order of their degrees to generate a sorted list of key-value pairs (node, degree) for each cluster. The ranking process operates in iterative rounds as follows:
   - For each cluster, the node with the highest degree is selected, appended to a temporary list, and removed from the cluster.
   - Nodes in the temporary list are sorted in descending order of their degrees.
   - This sorted list is appended to the overall ranking list R.
   - The process repeats until all nodes are ranked, ensuring that nodes with high degrees in the same cluster are given turns to be included in R over successive rounds.

Complexity Analysis

The time complexity of the LCD algorithm is mainly dominated by three parts: the computation of layers, clustering, and degree computation. Layer computation requires computing shortest paths from a given starting node which has a complexity of $O(n + m)$ using BFS for a graph with n nodes and m edges. The starting node can be discovered in $O(n + m)$ time. Clustering involves creating subgraphs and performing BFS traversal per layers. This step's complexity depends on the number of nodes and edges in each layer. If the number of layers is α, which is bounded by the graph's diameter, and each layer contains an average of n_L nodes, the clustering can be computed in $O(\alpha(n_L + m_L))$, where m_L represents the edges within each layer's subgraph. Finally, computing the degree of every node in every cluster requires no more than $O(m)$ in the worst case, since a node cannot simultaneously belong to multiple different clusters. Putting these steps together, the total complexity of the LCD algorithm is $O(n + m)$.

Performance Metrics

To evaluate the effectiveness of the proposed approach and compare it with existing methods for identifying influential nodes, we utilize three key metrics proposed in (Zhang *et al.*, 2016). The first metric, referred to as the infection scale $F(t)$, is used to measure the propagation of information diffusion within a network based on the *SIR* model. At any given time t, the infection scale is the total number of infected and recovered nodes in the system. This metric is considered an important indicator of the effectiveness of the spreader of selection algorithm. Mathematically, it is expressed as

$$F(t) = (n_{I(t)} + n_{R(t)})/n$$

Where $n_{I(t)}$ and $n_{R(t)}$ denote the number of infected and recovered nodes at time t, respectively. A larger $F(t)$ means more nodes have been affected by the initial influential nodes, while a shorter t indicates that these nodes propagate influence more quickly throughout the network. The second metric, called the final infected scale $F(t_c)$ which is used to measure the total number of nodes that have been affected when the system is stabilized. This metric considers all nodes that were infected and had recovered by the time t_c, at which there no infected nodes left in the network. It is calculated as

$$F(t_c) = n_{R(t_c)}/n$$

Where $n_{R(t_c)}$ denotes the number of recovered nodes at time t_c. A higher $F(t_c)$ value indicates a stronger spreading capacity of the initial nodes.

Finally, the average distance between spreaders, denoted as L_s, which is introduced to ensure that the selected spreaders are distributed across the network. If spreaders are too close together in certain areas, other parts of the network might be left undiscovered and so limiting the overall coverage. To prevent this, L_s is computed as the average shortest path length among the chosen spreaders, ensuring they are spread out as much as possible to improve information dissemination. It is formally defined as

$$L_s = \frac{1}{|S|(|S|-1)} \sum_{u,v \in S,\ u \neq v} d(u,v)$$

Where $d(u,v)$ denotes the shortest path distance between nodes u and v, and $|S|$ is the number of selected spreaders. Together, these metrics provide a comprehensive assessment of both the dynamic and structural aspects of the spreader selection process.

Algorithm 1: LCD Algorithm

Input: A network $G = (V, E)$, the number of spreaders κ, a starting node *start_node*.
Output: A ranking list R containing κ spreaders.
distances ← BFS(G, *start_node*);
eccentricity ← max(*distances.values*());
layers ← {i : [] | $i \in \{0, 1, \ldots, eccentricity\}$};
for (*node*, *dist*) ∈ *distances* **do**
  layers[*dist*].*append*(*node*);
clusters ← [];
for i, *layer* ∈ *layers* **do**
  if *layers*[i] = ∅ **then**
    continue;
  valid_nodes ← {*node* | $l \in \{i, i+1, \ldots, eccentricity\}$, *node* ∈ *layers*[$l$]};
  subgraph ← G.*subgraph*(*valid_nodes*);
  visited ← ∅;
  for *node* ∈ *layers*[i] **do**
    if *node* ∉ *visited* **then**
      cluster ← ∅;
      queue ← [*node*];
      while *queue* ≠ ∅ **do**
        current ← *queue*.*pop*(0);
        if *current* ∉ *visited* **then**
          visited ← *visited* ∪ {*current*};
          if *current* ∈ *layers*[i] **then**
            cluster ← *cluster* ∪ {*current*};
          for *neighbor* ∈ *subgraph.neighbors*(*current*) **do**
            queue.*append*(*neighbor*);
      clusters.*append*(*cluster*);
cluster_lists ← ∅;
for *cluster* ∈ *clusters* **do**
  sorted_cluster ← computer the degree for all nodes in cluster globally and sort cluster in descending order;
  cluster_lists.*append*(*sorted_cluster*);
while |R| < κ **do**
  round_candidates ← ∅;
  for *cluster* ∈ *cluster_lists* **do**
    if *cluster* ≠ ∅ **then**
      round_candidates.*append*(*cluster*.*pop*(0));
  if *round_candidates* ≠ ∅ **then**
    round_candidates ← Sort round_candidates in descending order;
    for *candidate* ∈ *round_candidates* **do**
      if |R| < κ **then**
        R.*append*(*candidate*);
return R;

3. EXPERIMENTAL RESULTS AND EVALUATION

To evaluate the performance of our proposed LCD method, we conducted a comparative analysis against several state-of-the-art approaches, including VoteRank (VR) (Zhang *et al*., 2016), VoteRank++ (VRP) (Liu *et al*., 2021), DegreeRank (DR) (Freeman *et al*., 2002), K-shell (KS) (Kitsak *et al*., 2010), H-index (HI) (Lu *et al*., 2016), ClusterRank (CR) (Chen *et al*., 2013), and EnRenew (En) (Guo *et al*., 2020). This comparison highlights the effectiveness of LCD in identifying influential spreaders and maximizing network influence under various conditions. A step-by-step demonstration of LCD is first presented using the Karate network to illustrate its functionality, providing a clear and structured example. This is followed by the introduction of the dataset and a presentation of the evaluation results.

An Example Network: Karate

In this section, we use the Karate Network as an example to explain the LCD method in detail. Figure 1 (a) depicts the Karate Network, which consists of 34 nodes and 78 edges, representing the relationships and interactions within a karate club. Each node corresponds to a member of the club, and edges indicate friendships or connections between members. Nodes highlighted in orange represent those with maximum eccentricity, where LCD method will select one of them as the starting node one. In this example, node 17 is chosen as the starting node.

From this starting point, we build layers in the network based on the shortest-path distances from node 17. Nodes directly connected to the starting node form the first layer, nodes connected to the first layer form the second layer, and so on. This approach creates a hierarchical structure, with nodes organized into layers that represent their relative distances from the starting point. Furthermore, nodes with the same color in Figure 1 (a) belong to the same layer, visually emphasizing the layered structure.

Within each layer, nodes are further grouped into clusters based on their connectivity patterns. Specifically, two nodes u and v belong to the same cluster if they are connected by a path using only nodes in the same layer or upper layers. Figure 1 (b) shows the identified clusters and their corresponding nodes. To enhance clarity, we show the identified clusters in layering tree in Figure 1 (b), a concept that has been introduce in (Chepoi and Dragan, 2000), which provides an abstract representation of these clusters and their hierarchical relationships. A layering tree, $T(G, s)$, of a graph G with respect to a layering partition is a graph whose nodes represent the clusters of the layering partition. Two nodes (clusters) C_1 and C_2 in the layering tree are connected by an edge if there exists a node $u \in C_1$ and a node $v \in C_2$ such that (u, v) is an edge in the original graph G. Colors in the layering tree correspond to clusters belonging to the same layer, illustrating the layered organization of the network. For instance, in layer three, node 12 is placed in cluster C_5 because there is no path connecting it to other nodes within the same layer (nodes in pink in Figure 1 (a)). It is important to note that the layering tree is not inherently part of the LCD method. However, we include it here to provide additional insight and clarity into the layered and clustered structure of the network.

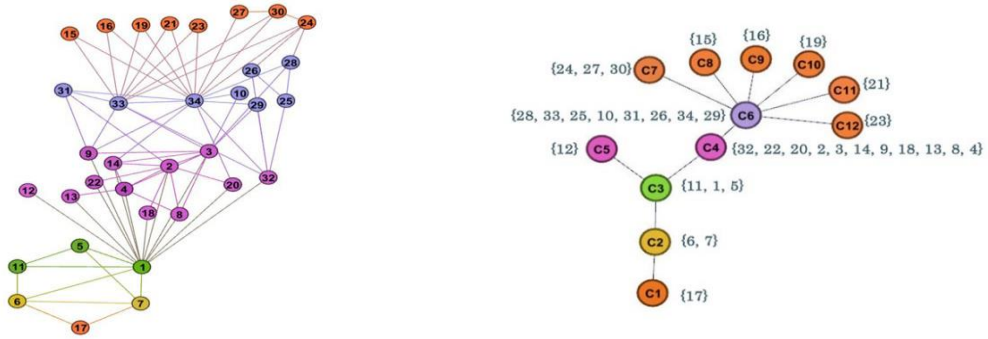
In the final step of the LCD algorithm, the degree of each node is computed across the entire network to ensure a global perspective on node influence. From each cluster, the node with the highest degree is selected and compiled into a temporary list S. The nodes in S are then sorted by their degrees in descending order and appended to the ranking list R. For instance, after the first round, R contains the top-ranked nodes [34, 1, 3, 24, 6, 17, 15, 16, 19, 21, 23, 12], representing the highest-degree nodes from the 12 clusters. This process iterates through subsequent rounds, selecting the highest-degree nodes from the remaining clusters. The final ranking list produced by the LCD method for the Karate Network is as follows: [34, 1, 3, 24, 6, 17, 15, 16, 19, 21, 23, 12, 33, 2, 7, 30, 11, 32, 31, 5, 27, 4, 28, 9, 26, 14, 25, 8, 29, 20, 10, 22, 18, 13]. This iterative ranking ensures comprehensive network coverage and minimizes redundancy, showcasing the significance of layered clustering. For instance, nodes 34 and 33 share 11 neighbors, that is $N(34) \cap N(33) = 11$, *meaning the influence of node 33 can largely be covered by*

node 34. Consequently, discarding node 33 and selecting another node with a moderate degree from another cluster ensures broader network impact.

In table 1, we present the top 5 nodes of the Karate network identified by each method tested in this paper, along with their corresponding $F(t)$ value, where $\lambda = 1.5$ and the source spreaders constitute 15% of the nodes (i.e., 5 nodes). The results are averaged over 100 independent runs. The methods included are DegreeRank (DR), ClusterRank (CR), VoteRank (VR), VoteRank++ (VRP), EnRenew (EN), K-shell (KS), H-Index (HI), and the LCD method.

Dataset Description

To show the performance of our algorithms, we have investigated nine networks with different sizes (small, medium, large) and different structural properties coming from different domains. For this analysis, all networks were treated as undirected and unweighted, and we considered only the largest connected component of each. Table 2 shows topological features of each network. Douban and Hamster networks are collected from the KONECT library (Kunegis, 2013) and Email, Router and CEnew are collected from the DIMACS (DIMACS). All other networks are available as part of the Stanford Large Network Dataset Collection (Stanford Large Network Dataset Collection (SNAP)).



a) Karate Network b) Layering Tree

***Figure 1**: Example Network: Karate.*

Table 1: *Top 5 nodes of the Karate network identified by each method and their corresponding F(t)*

Rank	DR	CR	VR	VRP	EN	KS	HI	LCD
1	34	8	34	34	1	34	1	34
2	1	14	1	1	34	1	3	1
3	33	4	33	33	33	33	14	3
4	3	9	3	2	3	3	33	24
5	2	16	2	6	2	2	34	6
$F(t)$	**0.340**	**0.378**	**0.349**	**0.374**	**0.347**	**0.346**	**0.343**	**0.383**

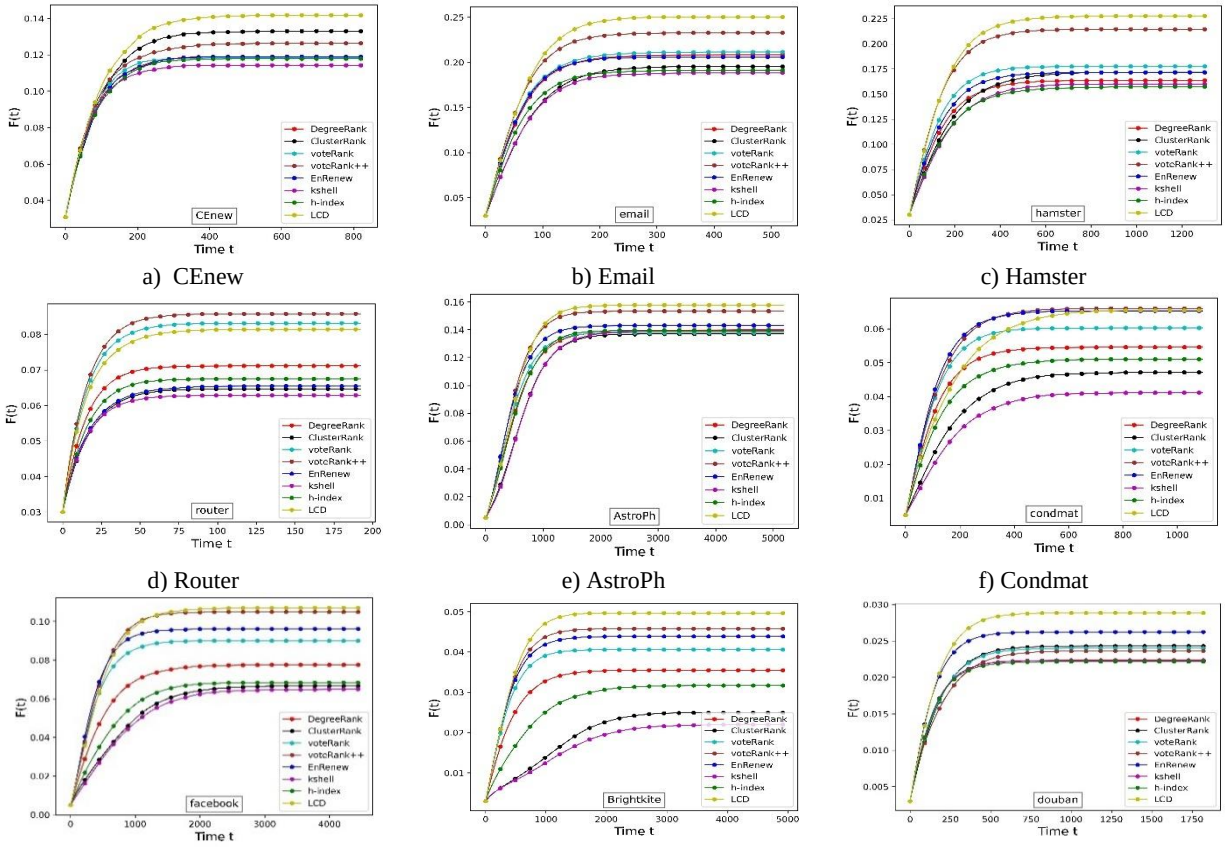
***Table 2**: Topological features of networks, where n and m are the total number of nodes and edges, respectively. $< k >$ is the average degree and k_{max} is the maximum degree. Clusters denote the number of clusters generated by LCD method. $\mu_c = \frac{<k>}{<k^2> - <k>}$ is the spreading threshold and $\mu = 1.5 * \mu_c$ is the infection probability.*

Network	n	m	$< k >$	k_{max}	clusters	μ_c	μ
CEnew	453	2025	8.94	237	66	0.0256	0.0384
Email	1133	5451	9.622	71	254	0.0565	0.0848
Hamster	2000	16098	16.098	237	391	0.0234	0.0351
Router	5022	6258	2.492	106	4557	0.0786	0.118
AstroPh	17903	196972	22.004	504	2774	0.0155	0.0232
CondMat	21363	91286	8.546	279	4740	0.0466	0.0699
Facebook	22470	170823	15.204	709	5448	0.0166	0.025
BrightKite	56739	212945	7.506	1134	29671	0.0159	0.0238
Douban	154908	327162	4.223	287	119779	0.0279	0.0418

Evaluation

Figure 2 illustrates the evolution of the infection scale $F(t)$ over time t using the *SIR* model, with results averaged across 100 independent runs at $\lambda = 1.5$. The experiments employ varying source spreader fractions p based on network size: 3% for smaller networks (CEnew, Email, Hamster, and Router), 0.5% for medium-sized networks (AstroPh, CondMat, and Facebook), and 0.3% for larger networks (BrightKite and Douban). Across most networks, the LCD method initiates faster information diffusion and achieves a larger final infection scale compared to other methods, as visually evident in the steeper initial slopes and higher plateaus of the LCD curves. While LCD generally exhibits greater variability in $F(t)$ compared to other methods, VoteRank++ is a notable exception, demonstrating consistently low variance in most networks. However, in the large Douban network, LCD surpasses VoteRank++ by 22% in the final affected scale ($F(t)$ of 0.0288 for LCD vs. 0.0236 for VoteRank++). It is worth noting that the VoteRank method is computationally expensive in large networks due to its high time complexity compared to all other methods tested. In the Router network, both VoteRank and VoteRank++ exhibit faster spreading rates and achieve larger final scales than LCD. Similarly, in the CondMat network, VoteRank++ and EnRenew initially spread information more rapidly than LCD, although all three methods ultimately reach comparable final infection scales ($F(t)$ values of 0.0660, 0.0655, and 0.0653 for VoteRank++, LCD, and EnRenew, respectively). Overall, Figure 2 demonstrates that LCD generally promotes both faster diffusion and a larger final infection scale compared to the other methods tested, highlighting its effectiveness in information spreading scenarios.

Figure 3 presents the final infected scale $F(t_c)$ of each method across all datasets with varying ratios p of source spreaders under the SIR model. The results clearly demonstrate the superior performance of our proposed method, LCD, across a diverse range of networks, including small, medium-sized, and large-scale graphs. The ratio of the source spreaders p is varied depending on the network size. It ranges from 0.010 to 0.045 in four small networks: CEnew, Email, Hamster, and Router. In mid-sized networks such as AstroPh, CondMat, and Facebook, p ranges from 0.0025 to 0.0055. For large-scale networks, including BrightKite and Douban, p ranges from 0.0005 to 0.0035. The results are averaged over 100 independent runs with infected rate $\lambda = 1.5$. Notably, LCD surpasses all other methods on seven out of nine networks across nearly all values of p, particularly excelling when the number of source spreaders is large. In networks such as CEnew, Email, Hamster, AstroPh, Facebook, BrightKite, and Douban, LCD consistently outperforms state-of-the-art methods like EnRenew, VoteRank, VoteRank++, k-Shell, and ClusterRank, achieving higher final affected scale $F(t_c)$ values across different source spreader ratios p. In the CondMat network, while EnRenew and VoteRank++ exhibit slightly better performance overall, LCD remains highly competitive. Similarly, in the Router network, VoteRank and VoteRank++ demonstrate superior results; however, LCD achieves comparable performance, nearly matching them at certain spreader ratios. These findings highlight LCD's robustness and adaptability to diverse network structures, ranging from sparse graphs to densely connected networks, further underscoring its effectiveness in identifying influential nodes and maximizing information spread.

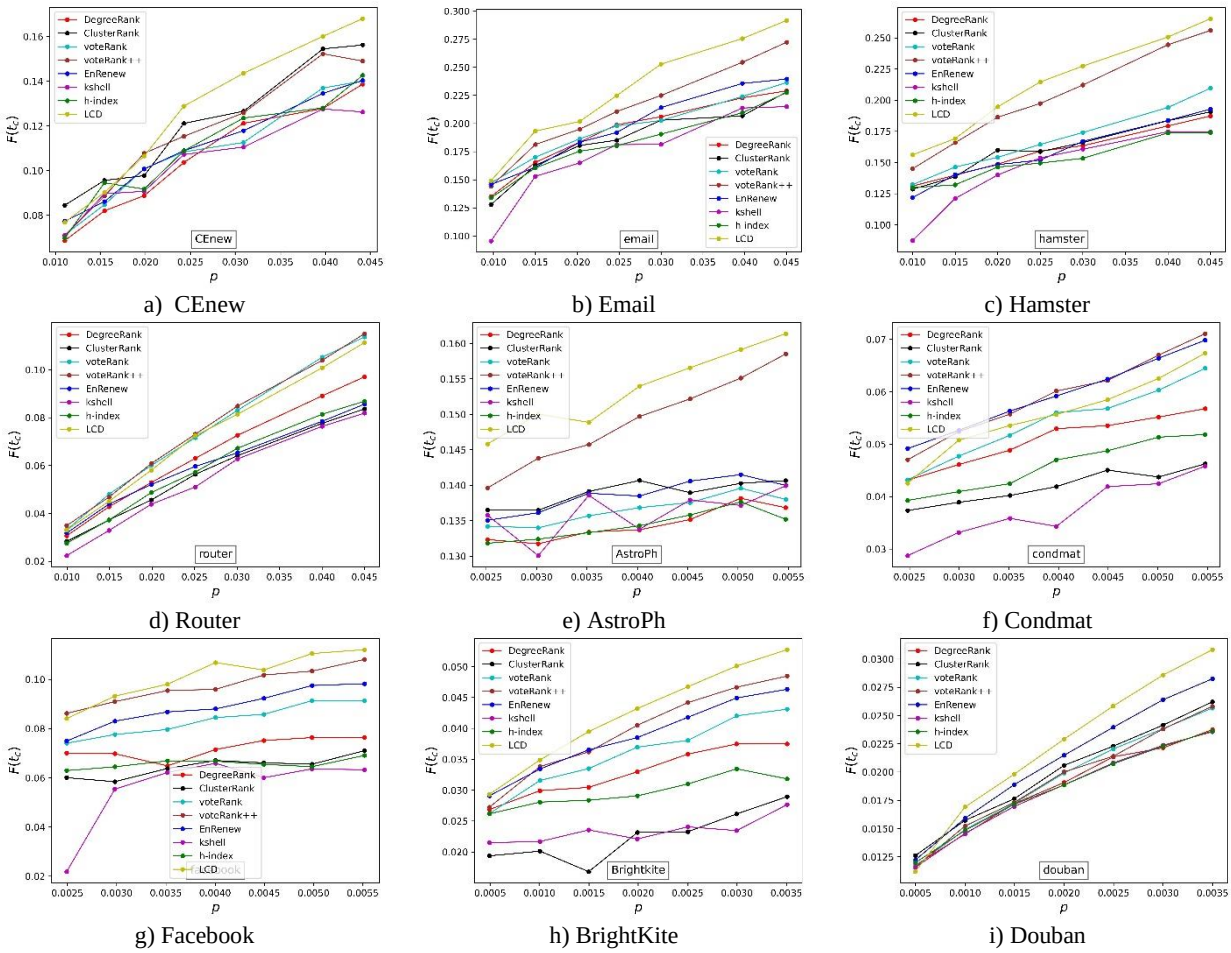


a) CEnew b) Email c) Hamster

d) Router e) AstroPh f) Condmat

g) Facebook h) BrightKite i) Douban

Figure 2*: The infected scale F(t) with the time t, where $\lambda = 1.5$ and the ratio p of source spreaders is 3% for CEnew, Email, Hamster and Router, 0.5% for AstropPh, Condmat, and Facebook, and 0.3% for BrightKite and Douban. The results are an average of 100 independent runs.*



a) CEnew b) Email c) Hamster

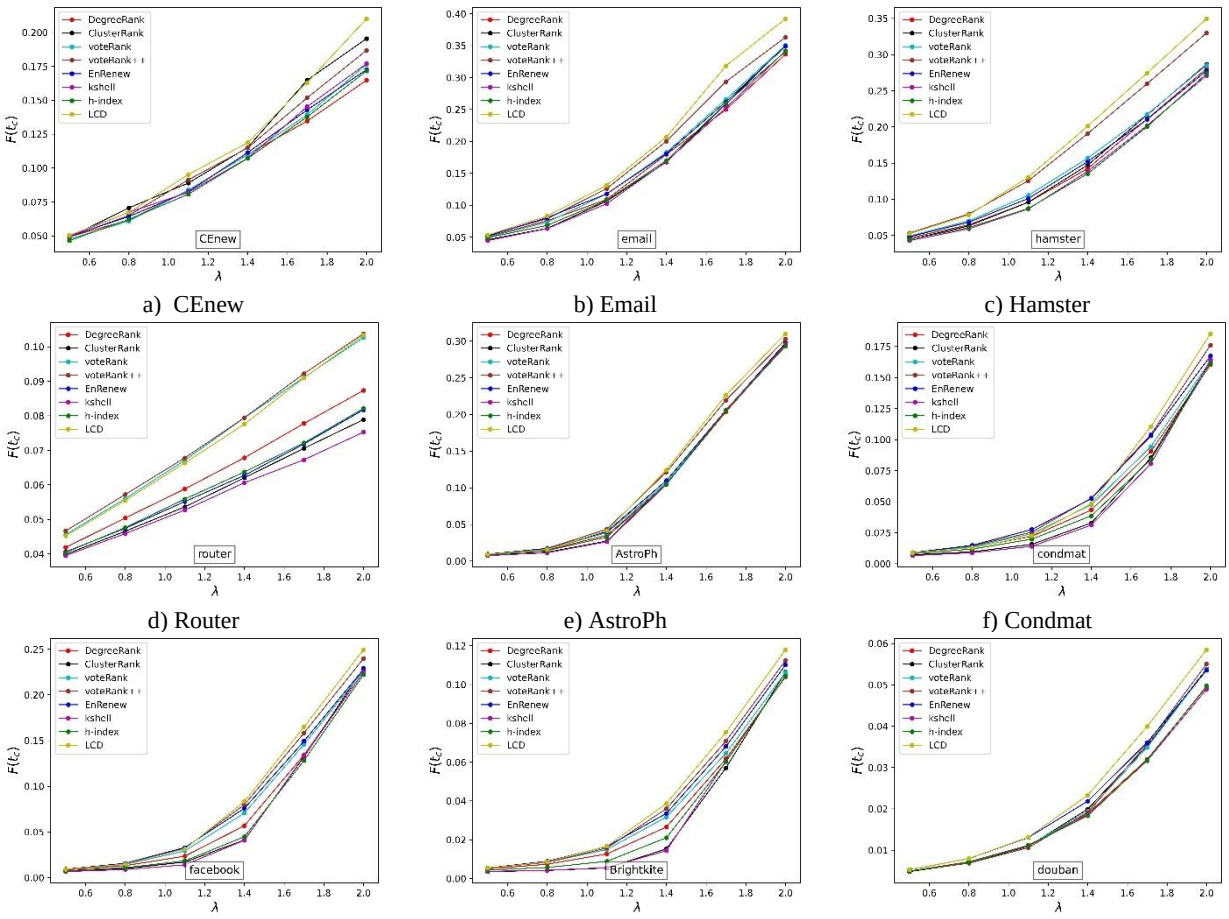
d) Router e) AstroPh f) Condmat

g) Facebook h) BrightKite i) Douban

Figure 3*: The final affected scale $F(t_c)$ with different ratio p of source spreaders. The results are obtained by averaging 100 independent runs with spread rate $\lambda = 1.5$ in SIR model.*

The infected rate λ plays a critical role in the propagation process, influencing how effectively information or viruses spread through networks. Therefore, Figure 4 illustrates the $F(t_c)$ values with varying λ across different methods on all networks, where the ratio p of source spreaders is 3% for CEnew, Email, Hamster, and Router, 0.5% for AstroPh, CondMat, and Facebook, and 0.3% for BrightKite and Douban. Results are averaged over 100 independent runs. It is evident that LCD consistently achieves broader spread scales across all networks, except for the Router network, which performs competitively with VoteRank and VoteRank++. When λ is too small, information fails to spread effectively, regardless of the chosen source spreaders. Conversely, if λ is too large, information spreads uncontrollably throughout the network. For this reason, λ is constrained between 0.5 and 2.0 to allow for a clear comparison of method performance (Zhang *et al*., 2016). This highlights LCD's superior generalization ability, particularly in scenarios with strong spreading conditions and higher λ values.

Distances between spreaders are pivotal in maximizing influence within a network, as spreaders that are more widely dispersed can impact a larger portion of the network (Hu *et al*., 2014). To confirm that the source spreaders identified by LCD are more broadly distributed compared to other methods, the average shortest path length (L_s) of spreaders detected by various approaches is examined. Figure 4 presents the L_s values across nine networks, revealing that LCD consistently achieves the highest L_s values across all networks, regardless of the ratio of source spreaders (p). This outcome demonstrates that LCD selects more dispersed spreaders, thereby maximizing network coverage. Furthermore, the gap between the L_s values of LCD and other methods is substantial, underscoring LCD's superiority in identifying well-scattered spreaders across diverse network structures. For instance, in the Email, Hamster, AstroPh, CondMat, Facebook, and BrightKite networks, the Ls value of LCD exceeds 1.5 times that of the best-performing method, VoteRank, signifying a remarkable improvement. In the CEnew and Router networks, LCD also outperforms other methods by a significant margin, particularly when p is small, highlighting its effectiveness in selecting well-distributed spreaders under limited conditions. This consistent performance across various networks and spreading scenarios underscores LCD's robustness and its ability to maximize influence by identifying source spreaders that are optimally distributed throughout the network.

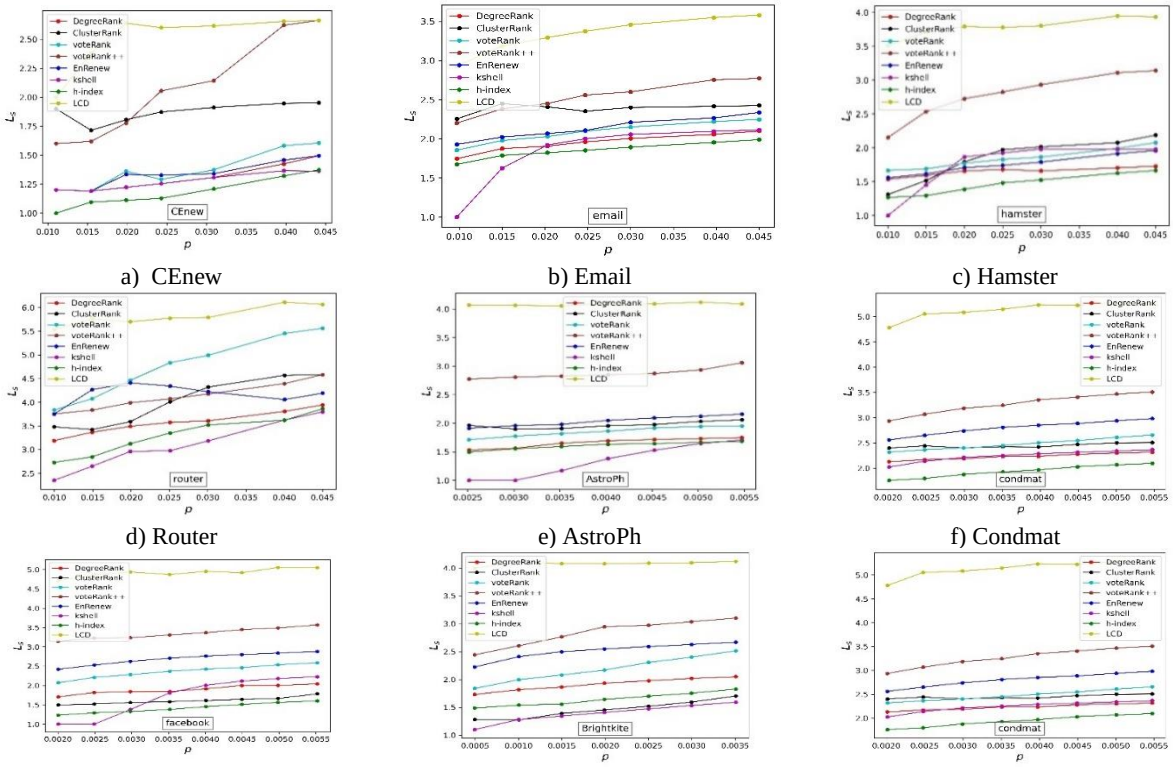


a) CEnew b) Email c) Hamster

d) Router e) AstroPh f) Condmat

g) Facebook h) BrightKite i) Douban

Figure 4: *The final affected scale $F(t_c)$ with different infected rate λ where the ratio p of source spreaders is 3% for CEnew, Email, Hamster and Router, 0.5% for AstropPh, Condmat, and Facebook, and 0.3% for BrightKite and Douban. The results are an average of 100 independent runs.*



a) CEnew b) Email c) Hamster

d) Router e) AstroPh f) Condmat

g) Facebook h) BrightKite i) Douban

Figure 5*: Average shortest path length L_s with different ratio p of source spreaders.*

Table 3: *The running time for each method in seconds. For VoteRank ++ (VRP), the reported time corresponds to processing only 2% of the nodes in the Douban network and 10% of the nodes in other networks.*

Network	DR	CR	VR	VRP	En	KS	HI	LCD
CEnew	0.01	0.04	0.07	0.42	0.03	0.01	0.01	0.07
Email	0.01	0.12	0.15	1.95	0.12	0.15	0.15	0.14
Hamster	0.01	0.31	0.61	10.03	0.32	0.031	0.06	0.43
Router	0.01	0.07	1.71	47.85	1.18	0.031	0.82	1.03
AstroPh	0.03	4.96	46.48	3560.84	49.42	0.56	0.53	13.41
CondMat	0.03	0.98	28.89	3407.59	22.17	0.2	0.17	6.79
Facebook	0.03	2.45	27.87	4353.51	24.5	0.37	0.29	10.17
BrightKite	0.04	3.11	179.31	16765.61	174.84	0.7	0.45	34.64
Douban	0.17	6.17	1649.15	57834.37	1213.25	1.34	0.79	275.14

Running Time

Table 3 compares the computational efficiency of several methods, including our proposed LCD algorithm, in terms of execution time (in seconds). DegreeRank (DR) consistently outperforms all methods in speed, requiring as little as 0.01 seconds for the smallest networks and a maximum of 0.17 seconds for the largest. This exceptional efficiency comes at the cost of lower accuracy and influence distribution. On the other hand, VoteRank, VoteRank++ (VRP) and EnRenew (En), while effective for certain tasks, exhibit significantly higher computational costs compared to our LCD method, particularly in larger networks. For instance, VoteRank++ requires an extensive 16,765.61 seconds to process just 10% of the BrightKite network nodes, and even more prohibitive 57,834.37 seconds to process only 2% of the Douban network nodes, underscoring its limited practicality for large-scale applications. Our LCD method demonstrates a balance between computational efficiency and effectiveness. It achieves moderate execution times, ranging from 0.07 seconds for smaller networks like CEnew to 275.14 seconds for larger networks such as Douban. While LCD is slightly slower than simpler methods such as DegreeRank (DR), K-shell (KS), and H-index (HI), it consistently outperforms them in identifying influential nodes. More importantly, LCD remains significantly faster than computationally intensive methods such as VoteRank (VR), VoteRank++ (VRP), and EnRenew, making it a scalable and practical option for real-world networks. All computations were performed on a system with an Intel Core i7 (2.60 GHz) processor and 16 GB of RAM.

CONCLUSION:

In this paper, we introduced the Layered Clustering Degree (LCD) method, an efficient approach for identifying influential nodes in complex networks. LCD integrates layering, clustering, and degree centrality to ensure maximum network coverage and balanced influence distribution. Through extensive experiments on nine real-world networks, LCD demonstrated better performance than several state-of-the-art algorithms, including VoteRank, K-shell, VoteRank++, ClusterRank, H-Index, EnRenew, and DegreeRank, in terms of computational complexity and spreading efficiency. The top-k influential nodes identified by LCD enable information to be propagated much more widely and faster than traditional degree centrality methods. While methods such as VoteRank++ demonstrate good spreading performance but suffer from high computational costs, LCD maintains better results with a computational complexity of $O(V + E)$, making it effective for large-scale networks. By addressing the trade-off between computational complexity and spreading efficiency, LCD offers a practical and scalable solution for identifying influential nodes in complex networks. One of the main limitations of the LCD algorithm is that it was designed for undirected networks and does not extend to weighted or directed networks. As many real-world networks, such as transportation or citation networks, have directions and edge weights, so extending LCD to such networks is open for future research. Additionally, future work cloud also explores optimizing the layering process and improving the clustering step to enhance LCD's performance. Furthermore, extending LCD's application to dynamic networks, where connections and node attributes evolve over time, could enable more adaptive and real-time analysis.

Ethical statement:

The Ethical Committee of the University of the Zakho, Duhok, Kurdistan region approved the current experiment.

Author Contributions:

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

Concept and design: Abdulhakeem O. Mohammed.

Acquisition, Analysis, or Interpretation of Data: Abdulhakeem O. Mohammed

Drafting of the Manuscript: Abdulhakeem O. Mohammed.

REFERENCES:

Anderson, R. M. and May, R. M. (1991). Infectious diseases of humans: dynamics and control. Oxford university press. DOI: https://doi.org/10.1093/oso/9780198545996.001.0001.

Bae, J. and Kim, S. (2014). Identifying and ranking influential spreaders in complex networks by neighborhood coreness. Physica A: Statistical Mechanics and its Applications, 395:549–559. DOI: https://doi.org/10.1016/j.physa.2013.10.047.

Buscarino, A., Fortuna, L., Frasca, M., and Latora, V. (2008). Disease spreading in populations of moving agents. Europhysics Letters, 82(3):38002. DOI: https://doi.org/10.1209/0295-5075/82/38002.

Castellano, C. and Pastor-Satorras, R. (2010). Thresholds for epidemic spreading in networks. Physical review letters, 105(21):218701. DOI: https://doi.org/10.1103/PhysRevLett.105.218701.

Centola, D. (2010). The spread of behavior in an online social network experiment. science, 329(5996):1194–1197. DOI: https://doi.org/10.1126/science.1185231.

Chen, D., L¨u, L., Shang, M.-S., Zhang, Y.-C., and Zhou, T. (2012). Identifying influential nodes in complex networks. Physica a: Statistical mechanics and its

applications, 391(4):1777–1787. DOI: https://doi.org/10.1016/j.physa.2011.09.017.

Chen, D.-B., Gao, H., Lu, L., and Zhou, T. (2013). Identifying influential nodes in large-scale directed networks: the role of clustering. PloS one, 8(10):e77455. DOI: https://doi.org/10.1371/journal.pone.0077455.

Chen, D.-B., Xiao, R., and Zeng, A. (2014). Predicting the evolution of spreading on complex networks. Scientific reports, 4(1):6108. DOI: https://doi.org/10.1038/srep06108.

Chen, W., Wang, Y., and Yang, S. (2009). Efficient influence maximization in social networks. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 199–208. DOI: https://doi.org/10.1145/1557019.1557047.

Chepoi, V. and Dragan, F. (2000). A note on distance approximating trees in graphs. European Journal of Combinatorics, 21(6):761–766. DOI: https://doi.org/10.1006/eujc.1999.0381.

DIMACS. The center for discrete mathematics and theoretical computer science. Available at: https://sites.cc.gatech.edu/dimacs10/archive/clustering.shtml.

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. Sociometry, pages 35–41. DOI: https://doi.org/10.2307/3033543.13

Freeman, L. C. et al. (2002). Centrality in social networks: Conceptual clarification. Social network: critical concepts in sociology. Londres: Routledge, 1:238–263. DOI: https://doi.org/10.1016/0378-8733(78)90021-7.

Guo, C., Yang, L., Chen, X., Chen, D., Gao, H., and Ma, J. (2020). Influential nodes identification in complex networks via information entropy. Entropy, 22(2):242. DOI: https://doi.org/10.3390/e22020242.

Hethcote, H. W. (2000). The mathematics of infectious diseases. SIAM review, 42(4):599–653. DOI: https://doi.org/10.1137/S0036144500371907.

Hu, Z.-L., Liu, J.-G., Yang, G.-Y., and Ren, Z.-M. (2014). Effects of the distance among multiple spreaders on the spreading. Europhysics Letters, 106(1):18002. DOI: https://doi.org/10.1209/0295-5075/106/18002.

Katz, L. (1953). A new status index derived from sociometric analysis. Psychometrika, 18(1):39–43. DOI: https://doi.org/10.1007/BF02289026.

Kitsak, M., Gallos, L. K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H. E., and Makse, H. A. (2010). Identification of influential spreaders in complex networks. Nature physics, 6(11):888–893. DOI: https://doi.org/10.1038/nphys1746.

Kumar, S. and Panda, B. (2020). Identifying influential nodes in social networks: Neighborhood coreness based voting approach. Physica A: Statistical Mechanics and its Applications, 553:124215. DOI: https://doi.org/10.1016/j.physa.2020.124215.

Kunegis, J. (2013). Konect: the koblenz network collection. In Proceedings of the 22nd international conference on world wide web, pages 1343–1350. Available at: http://konect.cc/networks/.

Liu, P., Li, L., Fang, S., and Yao, Y. (2021). Identifying influential nodes in social networks: A voting approach. Chaos, Solitons & Fractals, 152:111309. DOI: https://doi.org/10.1016/j.chaos.2021.111309.

Liu, Y., Tang, M., Zhou, T., and Do, Y. (2016). Identify influential spreaders in complex networks, the role of neighborhood. Physica A: Statistical Mechanics and its Applications, 452:289–298. DOI: https://doi.org/10.1016/j.physa.2016.02.028.

Lu, L., Chen, D.-B., and Zhou, T. (2011). The small world yields the most effective information spreading. New Journal of Physics, 13(12):123005. DOI: https://dx.doi.org/10.1088/1367-2630/13/12/123005.

Lu, L., Zhou, T., Zhang, Q.-M., and Stanley, H. E. (2016). The h-index of a network node and its relation to degree and coreness. Nature communications, 7(1):10168. DOI: https://doi.org/10.1038/ncomms10168.

Ouboter, T., Meester, R., and Trapman, P. (2016). Stochastic sir epidemics in a population with households and schools. Journal of Mathematical Biology, 72(5):1177–1193. DOI: https://doi.org/10.1007/s00285-015-0901-4.

Pastor-Satorras, R., Castellano, C., Van Mieghem, P., and Vespignani, A. (2015). Epidemic processes in complex networks. Reviews of modern physics, 87(3):925–979. DOI: https://doi.org/10.1103/RevModPhys.87.925.

Pastor-Satorras, R. and Vespignani, A. (2001). Epidemic spreading in scale-free networks. Physical review letters, 86(14):3200. DOI: https://doi.org/10.1103/PhysRevLett.86.3200.

Sabidussi, G. (1966). The centrality index of a graph. Psychometrika, 31(4):581–603. DOI: https://doi.org/10.1007/BF02289527.

Sheikhahmadi, A. and Nematbakhsh, M. A. (2017). Identification of multi-spreader users in social networks for viral marketing. Journal of Information Science, 43(3):412–423. DOI: https://doi.org/10.1177/0165551516644171.14.

Stanford Large Network Dataset Collection (SNAP). Stanford large network dataset. Available at: http://snap.stanford.edu/data/index.html.

Tao, Z., Zhongqian, F., and Binghong, W. (2006). Epidemic dynamics on complex networks. Progress in Natural Science, 16(5):452–457. DOI: https://doi.org/10.1080/10020070612330019.

Wang, F., Sun, Z., Gan, Q., Fan, A., Shi, H., and Hu, H. (2022). Influential node identification by aggregating local structure information. Physica A: Statistical Mechanics and its Applications, 593:126885. DOI: https://doi.org/10.1016/j.physa.2022.126885.

Wang, G., Alias, S. B., Sun, Z., Wang, F., Fan, A., and Hu, H. (2023). Influential nodes identification method based on adaptive adjustment of voting ability. Heliyon, 9(5). DOI: https://doi.org/10.1016/j.heliyon.2023.e16112.

Xu, G. and Dong, C. (2024). Cagm: A communicability-based adaptive gravity model for influential nodes identification in complex networks. Expert Systems with Applications, 235:121154. DOI: https://doi.org/10.1016/j.eswa.2023.121154.

Yang, Q., Wang, Y., Yu, S., and Wang, W. (2024). Identifying influential nodes through an improved k-shell iteration factor model. Expert Systems with Applications, 238:122077. DOI: https://doi.org/10.1016/j.eswa.2023.122077.

Yang, X. and Xiao, F. (2021). An improved gravity model to identify influential nodes in complex networks based on k-shell method. Knowledge-Based Systems, 227:107198. DOI: https://doi.org/10.1016/j.knosys.2021.107198.

Zeng, A. and Zhang, C.-J. (2013). Ranking spreaders by decomposing complex networks. Physics letters A, 377(14):1031–1035. DOI: https://doi.org/10.1016/j.physleta.2013.02.039.

Zhang, J.-X., Chen, D.-B., Dong, Q., and Zhao, Z.-D. (2016). Identifying a set of influential spreaders in complex networks. Scientific reports, 6(1):27823. DOI: https://doi.org/10.1038/srep27823.

Zhao, L., Cui, H., Qiu, X., Wang, X., and Wang, J. (2013). Sir rumor spreading model in the new media age. Physica A: Statistical Mechanics and its Applications, 392(4):995–1003. DOI: https://doi.org/10.1016/j.physa.2012.09.030.

Zhao, Z., Li, D., Sun, Y., Zhang, R., and Liu, J. (2023). Ranking influential spreaders based on both node k-shell and structural hole. Knowledge-Based Systems, 260:110163. DOI: https://doi.org/10.1016/j.knosys.2022.110163.