Nermin Kartli; Pelin Çetin; Selin Ayhan

New gravitational algorithms for the detection of overlapping and disjoint communities in weighted complex networks

Kybernetika, Vol. 61 (2025), No. 4, 509–536

Persistent URL: http://dml.cz/dmlcz/153073

KYBERNETIKA — VOLUME *61* (2025), NUMBER 4, PAGES 509–536

NEW GRAVITATIONAL ALGORITHMS FOR THE DETECTION OF OVERLAPPING AND DISJOINT COMMUNITIES IN WEIGHTED COMPLEX NETWORKS

Nermin Kartli, Pelin Çetin, and Selin Ayhan

The excessive increase in the amount of data caused a rise in the number of vertices and edges in the graph models. This gave rise to the concept of complex networks. Complex networks are present in almost every area of life, in social networks, natural sciences, drug discovery, etc. Detection of overlapping or disjoint communities in complex networks is an important problem. In this study, we propose two new algorithms to detect overlapping and disjoint communities in weighted complex networks. We assume that the weights represent how close the vertices are to each other in some sense. First, we calculate the similarity of the vertices to each other using the universal gravitational law, then place similar vertices in overlapping communities. Then, for each vertex in multiple communities, we calculate the attraction force of each community where this vertex is located. We leave the vertex in the community that attracts the vertex more and delete it from the others. The results of experiments conducted on complex networks consisting of real and artificial data show the efficiency of the proposed algorithms.

Keywords: weighted complex networks, community detection, overlapping community, disjoint community

Classification: 94C15

1. INTRODUCTION

An increasing number of studies are being published on complex networks, that is, mathematical graphs. Recently, researchers have begun to refer to graphs of enormous size as complex networks. One reason for this is that the research topic in these large graphs is different from classical graph theory problems. In complex networks, the aim is often to reveal hidden relationships and cluster according to a certain criterion. Complex networks can be encountered in many areas, such as social networks, drug discovery, natural language processing, etc. For example, a social media manager needs to be able to divide users into certain groups, expand the social network, suggest new friends or new jobs to the user, and send the same ads to similar people. These groups that we want to find in complex networks are called communities. There have been serious attempts to define the concept of community mathematically. Unfortunately, none of these definitions reflect real-life problems. Moreover, almost all of these definitions do not properly represent the concept of community in the verbal sense, even for some simple

DOI: 10.14736/kyb-2025-4-0509

graphs of small size. Various definitions of community are discussed in [3]. The lack of a precise mathematical definition of a community does not diminish the importance of searching for it, because in almost every field, it can be necessary to divide a complex network into communities, even when making decisions [36], calculating risks [1], and sorting [2, 39, 48].

Community detection problems can occur in all areas of life [4, 6, 7, 8, 12, 14, 17, 19, 20, 21, 22, 38, 43, 47, 53]. For this reason, this problem is addressed from different perspectives, and many studies are published every year. Since we cannot mention all of these studies, we will refer the works that are close to the approach we propose in this study. Nallusamy and Easwarakumar [35] propose a new community detection algorithm in a weighted undirected network. The algorithm determines which edges of the considered community should be replaced using the persistence loss parameter. Chin and Ratnavelu [13] present an improved version of the label propagation algorithm for community detection. He et al. [18] present a new method for calculating the analytical p-value, which indicates the statistical significance of an individual's belonging to a community in weighted networks. Hajibabaei et al. [16] propose a probabilistic generative model that estimates latent parameters and infers community structures based on the existence probabilities of weighted connections between edges. A multi-objective optimization approach is proposed for the problem of community detection in weighted networks by Samandari Masooleh et al. [44]. In this context, the authors adapt the whale optimization algorithm to work in parallel processing environments with discrete variable multi-criteria optimization problems. In the proposed method, the locations of individuals are discretized using a transfer function to transform continuous variables into discrete variables; the algorithm's initialization phase is restructured to prevent unrealistic relationships between variables; and the update phase is modified to ensure that the outputs are integers. A novel method for identifying clique structures in weighted complex networks is proposed by Goswami and Das [15]. In this method, each clique is assigned a density value, and two distinct interaction density thresholds are defined at both the individual and group levels to eliminate random or low-importance interactions during the clique formation process. Weighted maximum cliques of different sizes are used as fundamental building blocks, and a weighted version of the Jaccard similarity metric is developed to detect overlapping communities. This new metric is called the 'weighted Jaccard index.' A new algorithm based on a deep and sparse autoencoder is proposed Li et al. [27] by for community detection. In this study, firstly, a path weight matrix is generated by analyzing the second-degree neighborhood relationships of the nodes in the network. This matrix is combined with the weighted connection paths of each node to better represent the similarities between the nodes and their second-degree neighbors, resulting in a similarity matrix. Then, the deep sparse autoencoder developed with the unsupervised deep learning approach is used to produce a feature matrix that more comprehensively represents the structural properties of the network. This low-dimensional feature matrix is subjected to clustering with the K-means algorithm to reveal the community structure of the network. Kumar et al. [24] propose an improved neighbourhood Proximity-based Community Detection algorithm using Weighted Centrality for extracting overlapping community structures in undirected weighted networks. Kumar [25] designs a depth-first search-based algorithm for community detection in the weighted networks using the neighbourhood proximity measure. Prokop et al. [41] present a new algorithm for overlapping communities in weighted networks.

This algorithm detects hierarchical structures in overlapping networks. The authors use the interdependence between internal and external quality measures to evaluate the detected communities. Ma et al. [31] propose an algorithm based on the dissimilarity of nodes. Mohammed and Gunduc [33] propose a Transition Probability Matrix based on anonymous random walks as a new method for extracting the features of nodes in the graph. Suja et al. [49] propose a new optimization algorithm for overlapping community detection. Wang et al. [50] present a novel approach based on integrating spatial resistance and adjacency into the speaker-listener label propagation algorithm for uncovering overlapping communities in geospatial networks.

In this study, we define similarity based on the universal gravitational law and develop algorithms that can find overlapping and disjoint communities in weighted graphs based on this definition. The idea of using the gravitational force is used in many studies from different perspectives. Yang et al. [52] propose an algorithm based on the universal gravitation law to find overlapping community detections in unweighted networks, but in this algorithm, the force between nodes is calculated in a different way than in our study. Shen and Ma [45] improve the well-known label propagation algorithm for community detection in unweighted complex networks. The improvement is based on the law of gravity. Zalik and Zalik [55] formulate community detection in unweighted complex networks as an optimization problem facilitated by node attraction. The authors assume that each node is attracted by its neighbors. They propose an evolutionary community detection algorithm that uses modularity Q as a fitness function. Zhou et al. [56] propose a similarity based on the concepts of direct and indirect attraction force and design an algorithm based on the proposed attraction-force similarity. Sheng et al. [46] present a new community detection algorithm based on internode attraction. This algorithm starts from some nodes of the unweighted complex network and uses the gravitational relationship between nodes. Yu et al. [54] propose an approach called one-dimensional "gravity" (1DA) method for community detection in unweighted networks. The 1DA method uses the number of edges to measure 'gravity'. One of the works that proposes to use the gravity force is the study of Pattanayak et al. [40]. In this study, the authors present an innovative community detection algorithm that probabilistically estimates the size of communities using local structural information from randomly selected starting nodes. Subsequently, a gravity-based approach is applied to uncover communities around these nodes. The resulting local communities are integrated in a merging step to reveal the overall community structure of the entire network. Arasteh et al. [5] also introduce an algorithm for community detection inspired by the Gravitational law in an unweighted complex network. Lu and Dong [30] present a hierarchical gravity-based community detection algorithm to structure the supply chain network. The main idea of the algorithm is to characterize the overlapping conditions among communities by investigating the global gravity influence of focal firms. Li et al. propose [28] three improved label propagation algorithms based on the universal gravitation and give two methods (number of triangles and random walk algorithm with restart) that change the distance in the traditional physical sense to reduce the time complexity of our algorithms. The resulting attraction between nodes is used as the weight of the edge to propagate labels. Wu et al. [51] extract information about the network structure of a graph by introducing a new transition probability matrix based on Markov chains, which constructs a new graph. The authors then perform modularity optimization on the constructed one instead of the original one to detect disjoint communities. To detect overlapping

communities, they first initialize a cluster with a vital node as the attraction source, and then expand the cluster according to the graph attraction based on the transition probability. Chi et al. [11] propose a method for detecting overlapping communities in unweighted social networks. The method first creates initial communities by determining those that include the maximum local degree in the network. Then, if the attraction force between a community and a node in its vicinity exceeds a certain threshold value, this node is included in the relevant community. This process is repeated iteratively until the community structure is fixed. Nodes that have not yet been assigned to any community are considered overlapping nodes if they are attracted by more than one community above the threshold value. The membership degree of an overlapping node in any community is calculated by the ratio of the attraction force exerted by that community on the relevant node to the sum of all community attractions to which the node is exposed.

In this study, we consider weighted complex networks. We assume that the weight between two vertices indicates how close these vertices are to each other. For example, in a social media dataset, the closer a person A and person B are to each other, the less weight the edge connecting the vertices will have. In other words, we can say that the weights indicate the distance between two vertices. This condition regarding weights may not be met in some datasets; for example, in recommendation system problems, we can say that the more points a user gives to an item, the closer he/she feel to that item. In complex networks where the condition is met, we first apply the universal gravitation law and update the weights on the edges according to the attraction force exerted by the vertices at the edge ends. For each vertex, we take the degree of this vertex as its mass. Then we apply the modified Cetin and Emrah Amrahov algorithm [10] to the complex network we obtained. As a result of the algorithm, we find overlapping communities. Then, for each vertex in more than one community, we calculate the total attraction force from the vertices in these communities. We leave the vertex in the community that is more attracted by the community and delete it from the others. With this, we find disjoint communities. Since we obtain these communities differently, we find different communities from the Cetin and Emrah Amrahov study [9].

Recently, Khawaja et al. [23] proposed a common-neighbor-based overlapping community detection algorithm, building upon the work of Cetin and Emrah Amrahov, and Liu et al. [29]. Our differences from this work are as follows:

— We use our proposed gravitational similarity to calculate how close two vertices are to each other.

— We examine weighted graphs and therefore propose the weighted common neighbors criterion.

— We propose 2 algorithms to find both overlapping and disjoint communities.

2. PRELIMINARIES

2.1. Some definitions from the graph theory

Definition 2.1. Let $G = (V, E)$ be a simple undirected graph. If $uv \in E$ for a vertex $u \in V$, then v is called a neighbor of u. The set of all neighbors of a vertex u is denoted by $N(u)$.

Definition 2.2. The degree of a vertex u is defined as the number of its neighbors and is denoted by $degree(u)$, i. e., $degree(u) = |N(u)|$.

2.2. Cetin and Emrah Amrahov's overlapping community detection algorithm

Recently, Cetin and Emrah Amrahov developed a similar neighborhood-based algorithm to find overlapping communities in unweighted complex networks [10]. This algorithm is designed in 3 stages. In the first stage, each vertex is considered its neighbor, and the cosine similarity of neighboring vertices is calculated and assigned as the weight to the edge connecting these vertices. If $|V| = n$, then to calculate cosine similarity, an n-dimensional vector is assigned to each vertex as follows: For example, if a vertex u is adjacent to a vertex v_i, the ith coordinate of this vector is written as 1, otherwise 0. It is also assumed that each vertex is adjacent to itself. In the second stage, each vertex is initially assigned to a community consisting only of itself, and the edges are sorted in decreasing order of weight. Then, the edges are considered one by one in this order, and one end of the edge is added to one of the communities of the other according to a specific criterion called common neighbors. The criterion $CN(u, v)$, called common neighbors, is defined by the following formula:

$$CN(u, v) = \max_{C_v \in C} |C_v \cap n_u|. \tag{1}$$

Here, C is the set of communities formed up to the step under consideration, C_v is a community of the vertex v, and n_u is the set of all neighbors of the vertex u, including itself.

At this stage, if the vertices u and v for the edge (u, v) belong to singleton communities $\{u\}$ and $\{v\}$, these communities are merged into a community $\{u, v\}$, and $\{u\}$ and $\{v\}$ are removed from the community list. For edge (u, v), if the vertices u and v already belong to the same community, no action is taken, and the next edge is considered. Otherwise, $CN(u, v)$ is calculated to determine C_v*, and $CN(v, u)$ is calculated to determine C_u*, both maximize formula (1). If $CN(u, v*) > CN(v, u*)$, the vertex u is added to the community C_v*. If $CN(u, v*) < CN(v, u*)$, the vertex v is added to the community C_u*. In cases where multiple communities maximize formula (1), the first such community is selected. If $CN(u, v*) = CN(v, u*)$, the degrees of vertices u and v are compared. If the degrees are different, the vertex with the lower degree is added to the community of the vertex with the higher degree. If the degrees are equal, the vertex v is added to the community C_u*.

The third stage of the algorithm is the merging of the communities found. If more than half of the elements of one of the two communities considered are also elements of the other, or if one of the communities has only 2 elements and one of these elements belongs to the community of the other, these communities are merged into one community. The pseudo-code of the algorithm is given in Algorithm 1.

2.3. Evaluation metrics

To assess the quality of overlapping community detection results, we employ a set of widely used evaluation metrics, encompassing both topology-based and information-theoretic measures. These metrics enable a comprehensive evaluation ranging from

Algorithm 1: detectCommunities(G).

Input : $G = (V, E)$ is an unweighted simple graph
Output : C is the list of communities

/* Stage 1: Initialization */
for *each* $(u, v) \in E$ **do**
  // calculate the similarity for each pair of adjacent vertices
  $w(u, v) \leftarrow |N_u \cap N_v| / \sqrt{|N_u||N_v|}$
end
$C \leftarrow \phi$
for *each* $v \in V$ **do**
  // add a single vertex's community to the communities list
  $C \leftarrow C \cup \{v\}$
end
sort edges $(u, v) \in E$ according to $w(u, v)$ in descending order

/* Stage 2: Generating communities */
for *each edge* $(u, v) \in E$ **do**
  // if the vertices u and v have communities consisting only of themselves
  if $C_u = \{\{u\}\}$ *and* $C_v = \{\{v\}\}$ **then**
    // add $\{u, v\}$ to the community set and remove $\{u\}$ and $\{v\}$
    $C \leftarrow C \cup \{u, v\} - \{u\} - \{v\}$
  end
  else if *there are communities* $C_u \in C$ *and* $C_v \in C$ *such that* $v \notin C_u$ *and* $u \notin C_v$ **then**
    $CN(u, v) \leftarrow \max_{C_v \in C} |C_v \cap n_u|$
    $C_v^* \leftarrow arg\,max\ CN(u, v)$
    $CN(v, u) \leftarrow \max_{C_u \in C} |C_u \cap n_v|$
    $C_u^* \leftarrow arg\,max\ CN(v, u)$
    if $CN(u, v) > CN(v, u)$ *or* $(CN(u, v) \leftarrow CN(v, u)$ *and* $deg(u) < deg(v))$ **then**
      $C \leftarrow C - C_v^*$
      $C_v^* \leftarrow C_v^* \cup \{u\}$
      $C \leftarrow C \cup C_v^*$
      if $\{u\} \in C$ **then**
        $C \leftarrow C - \{u\}$
      end
    end
    else
      $C \leftarrow C - C_u^*$
      $C_u^* \leftarrow C_u^* \cup \{v\}$
      $C \leftarrow C \cup C_u^*$
      if $\{v\} \in C$ **then**
        $C \leftarrow C - \{v\}$
      end
    end
  end
end

/* Stage 3: Merging communities */
// $C = \{C_1, C_2, ...C_k\}$
$k \leftarrow length(C)$
sort the set C in according to the numbers of elements in the communities C_i in descending order
$l \leftarrow 1$
for $i = 2$ *to* k **do**
  for $j = l$ *downto* 1 **do**
    if $|C_i \cap C_j| > |C_i|/2$ *or* $(|C_i| = 2$ *and* $|C_i \cap C_j| = 1)$ **then**
      $C \leftarrow C - C_j - C_i$
      $C_i \leftarrow C_i \cup C_j$
      $C \leftarrow C \cup C_i$
    end
  end
  renumber communities in list C with indices less than or equal to i and assign the number of the renumbered communities to l
end
return C

structural cohesion to alignment with ground-truth communities. In our experiments, we utilize the implementations of Overlapping Normalized Mutual Information (ONMI_{LFK}), Omega Index, and average F1-score as provided by the `CDlib` library.

2.3.1. Fuzzy overlapping modularity (Q_{oc})

In this study, we evaluated the structural quality of overlapping communities using a fuzzy modularity measure used by Chi, Qu, and Fu [11]. This measure, denoted as Q_{oc}, extends classical modularity to overlapping community structures by incorporating the degree of shared membership between node pairs across communities. Q_{oc} metric is defined as:

$$Q_{oc} = \frac{1}{2m} \sum_{i,j \in V} \left[A_{ij} - \frac{d_i d_j}{2m} \right] \cdot \max_{c \in C} \left(\min(\mu_c(i), \mu_c(j))^{1/2} \right).$$

Here, A_{ij} denotes the edge weight between the nodes i and j, while d_i is the weighted degree of the node i, and m is the total edge weight in the network. The function $\mu_c(i) \in [0, 1]$ represents the fuzzy membership strength of node i in the community c, allowing partial affiliation with multiple communities.

Since community input is in the form of node sets, where each node can belong to multiple communities without specified degree of membership, we automatically generate fuzzy memberships by distributing the affiliation of each node uniformly between its communities. That is, if a node belongs to k communities, its membership in each is set to $\mu = \frac{1}{k}$.

This formulation allows the metric to assign greater weight to node pairs that are more strongly coaffiliated in shared communities, making it more sensitive to both network structure and overlapping topology.

2.3.2. Overlapping normalized mutual information (ONMI_{LFK})

Overlapping Normalized Mutual Information (ONMI_{LFK}) is a generalization of the classical Normalized Mutual Information (NMI) metric, designed to evaluate the similarity between overlapping community structures. Proposed by Lancichinetti et al. [26], it extends mutual information to settings where nodes can belong to multiple communities simultaneously.

Unlike traditional NMI, which assumes disjoint partitions, ONMI_{LFK} accommodates multi-membership by considering the distribution of community assignments across nodes. Quantifies the reduction in uncertainty of one partition given the knowledge of the other, thereby capturing the mutual dependence between the two.

A score of 1 indicates identical community assignments, while a score of 0 suggests no informational overlap. This metric is particularly suitable for evaluating the alignment between detected overlapping communities and a reference point of ground truth.

A binary vector is defined for each node. For example, for X:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{iK}), \quad \text{here } x_{ik} \in \{0, 1\}.$$

Similarly, for Y:

$$y_i = (y_{i1}, y_{i2}, \ldots, y_{iL}), \quad y_{il} \in \{0, 1\}.$$

These vectors indicate which communities each node belongs to.

Mutual information, based on information theory, is used to compare community structures.

$$I(X;Y) = \sum_{x,y} P(x,y) \log \left(\frac{P(x,y)}{P(x)P(y)} \right).$$

Here:

- $P(x)$: The probability that a node has a ground-truth vector x.
- $P(y)$: The probability that a node has a ground-truth vector y.
- $P(x,y)$: The probability that a node has both the vectors x and y.

These probabilities are estimated on the basis of the distribution of nodes.

Entropy is calculated as follows:

$$H(X) = -\sum_{x} P(x) \log P(x), \quad H(Y) = -\sum_{y} P(y) \log P(y).$$

As a result, the **Overlapping Normalized Mutual Information** metric is defined as follows:

$$ONMI(X,Y) = \frac{I(X;Y)}{\max(H(X), H(Y))}.$$

- $ONMI \in [0,1]$.
- $ONMI = 1$ means that the structures are exactly the same.
- $ONMI = 0$ means that the structures are completely different.
- It can be applied to both overlapping and disjoint communities.

2.3.3. F-score

The F-score is a classical metric from information retrieval, combining precision and recall into a single measure. In the context of community detection, it evaluates the correctness of node assignments relative to the ground truth. It is computed as:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where precision is the fraction of correctly predicted nodes among those assigned to a community, and recall is the fraction of ground-truth community members correctly identified. It computes the optimal one-to-one matches between the communities of the two partitions and reports the mean F-score across all matches. This approach is robust to both overlapping and non-overlapping, complete or partial community structures [42].

2.3.4. Omega index

The Omega Index is a pair-counting metric that quantifies the similarity between two overlapping community partitions by comparing how many communities each pair of nodes shares in both partitions. It extends the Adjusted Rand Index to overlapping clusters by adjusting for chance agreement.

Given two partitions A and B on n nodes, the Omega Index is calculated as:

$$\Omega = \frac{\text{Observed agreement} - \text{Expected agreement}}{\text{Maximum agreement} - \text{Expected agreement}}$$

where the observed agreement counts node pairs sharing the same number of communities in both partitions, and the expected agreement models the agreement by chance. Formally,

$$\text{Observed agreement} = \sum_{i<j} \mathbb{I}(s_{ij}^A = s_{ij}^B)$$

with s_{ij}^A and s_{ij}^B denoting the number of shared communities between nodes i and j in partitions A and B, respectively, and $\mathbb{I}$ the indicator function.

A higher Omega Index indicates greater similarity between the overlapping community structures. This metric was introduced by Murray et al. [34] for evaluating clustering similarity in overlapping settings.

3. PROPOSED ALGORITHMS

3.1. Proposed overlapping community detection algorithm

In this subsection, we propose a new gravitational law-based algorithm to detect overlapping communities in complex weighted networks. Our proposed algorithm uses some of the same steps as the recently proposed algorithm by Cetin and Emrah Amrahov [10] to find overlapping communities in unweighted complex networks. Unlike Cetin and Emrah Amrahov's algorithm, the input of our proposed algorithm is a weighted graph; we use gravitational similarity instead of cosine similarity, and we propose to use another function instead of the $CN(u, v)$ function. We also modify our algorithm in the next subsection and propose a new algorithm that can also find disjoint communities.

The first idea that comes to mind to adapt the first stage of Algorithm 1 to weighted complex networks is straightforward. Let us remove lines 1-3 of Algorithm 1 and use the given weights instead. Since the weights represent distances in our example, we use ascending order to sort the edges. The modified version of the first stage is given below. We implemented the remaining parts consistently and referred to this version as Algorithm 1. Let us call the algorithm obtained by changing the first stage of Algorithm 1 as in Algorithm 2, as Algorithm 1A. With a simple example, let us show that it is not a good algorithm.

Algorithm 2: Modified version of the first stage of detectCommunities(G).

Input : $G = (V, E)$ is a weighted simple graph
Output : C is the list of communities

```
/* Stage 1:  Initialization                                          */
C ← φ
for each v ∈ V do
    // add a single vertex's community to the communities list
    C ← C ∪ {v}
end
sort edges (u, v) ∈ E according to w(u, v) in ascending order
```

Example 3.1. The input graph is given in Figure 1. Let us assume that this graph corresponds to social media. The weights indicate the closeness of two vertices to each other. The lower the weight between two vertices, the closer these two vertices are to each other.

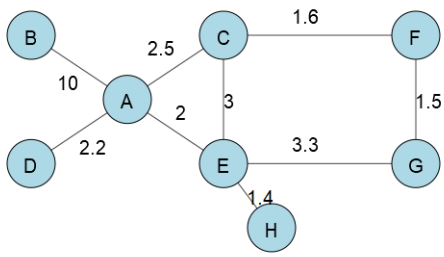

Fig. 1. Input Weighted Graph.

Thus, after the first 2 stages of Algorithm 1A, the following 4 communities are obtained:

$$C_1 = \{A, B, C, D, E, G\}$$
$$C_2 = \{C, F\}$$
$$C_3 = \{E, H\}$$
$$C_4 = \{F, G\}.$$

After the third stage of Algorithm 1A, a single community is formed for this graph that includes all the vertices of the graph. In other words, if we use the given weights directly, the entire graph forms a community.

3.1.1. Gravitational similarity

Consider a gravitational force between the vertices of the graph, similar to Newton's universal law of gravitation. Let us assume that the mass of a vertex $m(u)$ is its degree, and the distance r between the vertices is the weight of the edge connecting these vertices. A vertex generally contains within a collection the vertex's neighbors (all or

Step	(u,v)	w(u,v)	CN(u,v)	CN(v,u)	Vertex with higher degree	Communities
0						{A},{B},{C},{D},{E},{F},{G},{H}
1	EH	1.2				{A},{B},{C},{D},{F},{G},{E,H}
2	FG	1.5				{A},{B},{C},{D},{E,H},{F,G}
3	CF	1.6	1	1	eq.	{A},{B},{C,F},{D},{E,H},{F,G}
4	AE	2	1	1	eq.	{A,E},{B},{C,F},{D},{E,H},{F,G}
5	AD	2.2	1	1	A	{A,D,E},{B},{C,F},{E,H},{F,G}
6	AC	2.5	1	2		{A,C,D,E},{B},{C,F},{E,H},{F,G}
7	CE	3				{A,C,D,E},{B},{C,F},{E,H},{F,G}
8	EG	3.5	1	1	E	{A,C,D,E,G},{B},{C,F},{E,H},{F,G}
9	AB	10	1	1	A	{A,B,C,D,E,G},{C,F},{E,H},{F,G}

Tab. 1. Applying the first and the second stages of Algorithm 1A to Example 3.1 with given weights.

some of them) or the neighbors of its neighbors. The degree of a vertex is equal to the number of its neighbors, as defined above. When considering two adjacent vertices, it is natural to expect the vertex with the higher degree to attract the vertex with the lower degree to its own collection, just as in the gravitational law, the more massive object attracts the less massive object. Therefore, in our proposed method, it is natural to define the degree of a node as its mass. Thus, the gravitation force between adjacent vertices u and v is calculated by the following formula:

$$F(u,v) = \gamma \frac{deg(u)deg(v)}{w(u,v)^2}. \tag{2}$$

Here γ is the gravitational constant. Since the result of the algorithm we propose in this study is independent of the value of this constant, we take $\gamma = 1$ throughout the paper.

Algorithm 3: Modified version of the first stage of detectOverlappingCommunities(G) with gravitational forces.

Input : $G = (V, E)$ is a weighted simple graph and $w(u, v)$ are the weights
Output : C is the list of communities

```
/* Stage 1:  Initialization                                          */
for each (u, v) ∈ E do
    // calculate the gravitational weights for each edge
    F(u, v) ← deg(u)deg(v)/w(u, v)^2
end
C ← φ
for each v ∈ V do
    // add a single vertex's community to the communities list
    C ← C ∪ {v}
end
sort edges (u, v) ∈ E according to F(u, v) in descending order
```

We calculate the gravitational force between vertices u and v for all (u, v) in the given complex network, using the formula 2 and replacing the weight of this edge with the found value.

Now, we change the first stage of Algorithm 1 to Algorithm 3. We applied the remaining stages consistently. Let's call the obtained algorithm Algorithm 1B. Applying Algorithm 1B, we show in Table 2 step by step.

Step	(u,v)	F(u,v)	CN(u.v)	CN(v,u)	Vertex with higher degree	Communities
0						{A},{B},{C},{D},{E},{F},{G},{H}
1	AE	4				{B},{C},{D},{F},{G},{H},{A,E}
2	EH	2.78				{B},{C},{D},{F},{G},{H},{A,E,H}
3	CF	2.34	1	1	C	{B},{D},{G},{A,E,H},{C,F}
4	AC	1.92	1	2		{B},{D},{G},{A,C,E,H},{C,F}
5	FG	1.77	1	1	equal	{B},{D},{A,C,E,H},{C,F,G}
6	CE	1.33				{B},{D},{A,C,E,H},{C,F,G}
7	AD	0.83	1	1	A	{B},{A,C,D,E,H},{C,F,G}
8	EG	0.73	2	1		{B},{A,C,D,E,H},{C,E,F,G}
9	AB	0.04	1	1	A	{A,B,C,D,E,H},{C,E,F,G}

Tab. 2. Applying first and second stages of Algorithm 1B to Example 3.1 with gravitational weights.

After applying the first and second stages of Algorithm 1B we obtain 2 overlapping communities below:

$$C_1 = \{A, B, C, D, E, H\}$$

$$C_2 = \{C, E, F, G\}.$$

The number of elements C_2 in common with C_1 is 2, which is not more than half of the number of elements in C_2, so there is no merge. As a result, the algorithm finds the communities $C_1 = \{A, B, C, D, E, H\}$ and $C_2 = \{C, E, F, G\}$. We show these communities in Fig 2.

3.1.2. Common neighbors weighted

Let us now discuss the formula 1. This criterion was proposed for unweighted complex networks in [10]. Since all edges are equivalent in weight for unweighted complex networks, it makes sense to use the formula 1, but the situation is different for weighted complex networks. We propose to use the function $CNW(u, v)$, which we will define below, instead of the function $CN(u, v)$ in Algorithm 1B. The name of the function $CNW(u, v)$ is an abbreviation of "Common Neighbors Weighted".

For a community C_v, if the intersection of this community with the set n_u is nonempty, then the vertex u is connected to some vertices in the set C_v. We define gravitational

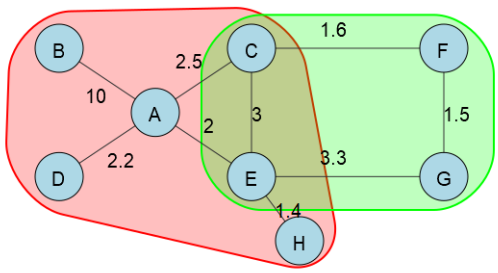

Fig. 2. Overlapping Communities after Applying Algorithm 1B to Example 3.1 with gravitational weights.

weight $F_u(C_v)$ as the function that returns the sum of the gravitational forces between u and x, when the vertex x varies in the set $C_v \cap n_u$.

$$F_u(C_v) = \sum_{x \in C_v \cap n_u} F(u, x). \tag{3}$$

Let C denote the set of all existing communities in the current step. We define the function $CNW(u, v)$ that returns the maximum gravitational weight $F_u(C_v)$ among all $C_v \in C$:

$$CNW(u, v) = \max_{C_v \in C} F_u(C_v). \tag{4}$$

We propose to use the function $CNW(u, v)$ as the criterion in the second stage of Algorithm 1. Consequently, we obtain a new algorithm to find overlapping communities in complex networks whose weights imply the closeness of the vertices to each other. The pseudocode of this proposed algorithm is given in Algorithm 4.

The following example demonstrates the difference between using $CN(u, v)$ or $CNW(u, v)$ as criteria.

Example 3.2. The input graph is given in Figure 3. Let us assume that this graph corresponds to social media. Weights indicate the closeness of two vertices to each other. The lower the weight between two vertices, the closer these two vertices are to each other.

Solution 1 (Detection of overlapping communities using Algorithm 1B in Example 3.2 with gravitational weights) The step-by-step application of the first and second stages of Algorithm 1B is shown in Table 3.

Algorithm 4: gravitationalOverlappingCommunityDetection(G,w).

Input : $G = (V, E)$ is a weighted simple graph and $w(u, v)$ are the weights
Output : C is the list of communities

/* Stage 1: Initialization */
for *each* $(u, v) \in E$ **do**
  // calculate the gravitational weights for each edge
  $F(u, v) \leftarrow deg(u)deg(v)/w(u, v)^2$
end
$C \leftarrow \phi$
for *each* $v \in V$ **do**
  // add a single vertex's community to the communities list
  $C \leftarrow C \cup \{v\}$
end
sort edges $(u, v) \in E$ according to $F(u, v)$ in descending order
/* Stage 2: Generating communities */
for *each edge* $(u, v) \in E$ **do**
  // if the vertices u and v have communities consisting only of themselves
  if $C_u = \{\{u\}\}$ *and* $C_v = \{\{v\}\}$ **then**
    // add $\{u, v\}$ to the community set and remove $\{u\}$ and $\{v\}$
    $C \leftarrow C \cup \{u, v\} - \{u\} - \{v\}$
  end
  else if *there are communities* $C_u \in C$ *and* $C_v \in C$ *such that* $v \notin C_u$ *and* $u \notin C_v$ **then**
    $CNW(u, v) \leftarrow \max_{C_v \in C} F_u(C_v)$
    $C_v^* \leftarrow arg\,max\ CNW(u, v)$
    $CNW(v, u) \leftarrow \max_{C_u \in C} F_v(C_u)$
    $C_u^* \leftarrow arg\,max\ CNW(v, u)$
    if $CNW(u, v) > CNW(v, u)$ *or* $(CNW(u, v) = CNW(v, u)$ *and* $deg(u) < deg(v))$ **then**
      $C \leftarrow C - C_v^*$
      $C_v^* \leftarrow C_v^* \cup \{u\}$
      $C \leftarrow C \cup C_v^*$
      if $\{u\} \in C$ **then**
        $C \leftarrow C - \{u\}$
      end
    end
    else
      $C \leftarrow C - C_u^*$
      $C_u^* \leftarrow C_u^* \cup \{v\}$
      $C \leftarrow C \cup C_u^*$
      if $\{v\} \in C$ **then**
        $C \leftarrow C - \{v\}$
      end
    end
  end
end
/* Stage 3: Merging communities */
// $C = \{C_1, C_2, ...C_k\}$
$k \leftarrow length(C)$
sort the set C in according to the numbers of elements in the communities C_i in descending order
$l \leftarrow 1$
for $i = 2$ *to* k **do**
  for $j = l$ *downto* 1 **do**
    if $|C_i \cap C_j| > |C_i|/2$ *or* $(|C_i| = 2$ *and* $|C_i \cap C_j| = 1)$ **then**
      $C \leftarrow C - C_j - C_i$
      $C_i \leftarrow C_i \cup C_j$
      $C \leftarrow C \cup C_i$
    end
  end
  renumber communities in list C with indices less than or equal to i and assign the number of the renumbered communities to l
end
return C

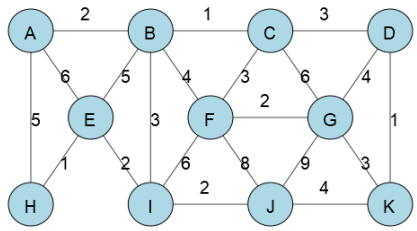

Fig. 3. Input Weighted Graph.

Step	(u,v)	F (u,v)	CN (u,v)	CN (v,u)	Vertex with higher degree	Communities
0						{A},{B},{C},{D},{E},{F},{G},{H},{I},{J},{K}
1	BC	20				{A},{D},{E},{F},{G},{H},{I},{J},{K},{B,C}
2	DK	9				{A},{E},{F},{G},{H},{I},{J},{B,C},{D,K}
3	EH	8				{A},{F},{G},{I},{J},{B,C},{D,K},{E,H}
4	FG	6.25				{A},{I},{J},{B,C},{D,K},{E,H},{F,G}
5	EI	4	1	1	eq.	{A},{J},{B,C},{D,K},{E,H,I},{F,G}
6	IJ	4	1	1	eq.	{A},{B,C},{D,K},{E,H,I,J},{F,G}
7	AB	3.75	1	1	B	{A,B,C},{D,K},{E,H,I,J},{F,G}
8	BI	2.22	2	1		{A,B,C},{D,K},{B,E,H,I,J},{F,G}
9	CF	2.22	2	2	F	{A,B,C},{D,K},{B,E,H,I,J},{C,F,G}
10	GK	1.67	2	1		{A,B,C},{D,K,G},{B,E,H,I,J},{C,F,G}
11	BF	1.56	2	3		{A,B,C},{D,K,G},{B,E,F,H,I,J},{C,F,G}
12	CD	1.33	2	2	C	{A,B,C},{D,K,G},{B,E,F,H,I,J},{C,D,F,G}
13	DG	0.94				{A,B,C},{D,K,G},{B,E,F,H,I,J},{C,D,F,G}
14	BE	0.8				{A,B,C},{D,K,G},{B,E,F,H,I,J},{C,D,F,G}
15	JK	0.75	2	1		{A,B,C},{J,D,G,K},{B,E,F,H,I,J},{C,D,F,G}
16	CG	0.56				{A,B,C},{J,D,G,K},{B,E,F,H,I,J},{C,D,F,G}
17	FI	0.56				{A,B,C},{J,D,G,K},{B,E,F,H,I,J},{C,D,F,G}
18	AE	0.33	3	2		{A,B,C},{J,D,G,K},{A,B,E,F,H,I,J},{C,D,F,G}
19	FJ	0.31				{A,B,C},{J,D,G,K},{A,B,E,F,H,I,J},{C,D,F,G}
20	GJ	0.25				{A,B,C},{J,D,G,K},{A,B,E,F,H,I,J},{C,D,F,G}
21	AH	0.24				{A,B,C},{J,D,G,K},{A,B,E,F,H,I,J},{C,D,F,G}

Tab. 3. Application of the first and the second stages of Algorithm 1B to Example 3.2 with gravitational weights.

After applying the first 2 stages of the algorithm, we obtain 4 communities:

$$C_1 = \{A, B, E, F, H, I, J\}$$
$$C_2 = \{D, G, J, K\}$$
$$C_3 = \{C, D, F, G\}$$
$$C_4 = \{A, B, C\}.$$

After the third, merging stage of the algorithm, we have 3 communities:

$$C_1 = \{A, B, C, E, F, H, I, J\}$$
$$C_2 = \{J, D, G, K\}$$
$$C_3 = \{C, D, F, G\}.$$

The communities found are shown in Figure 4.

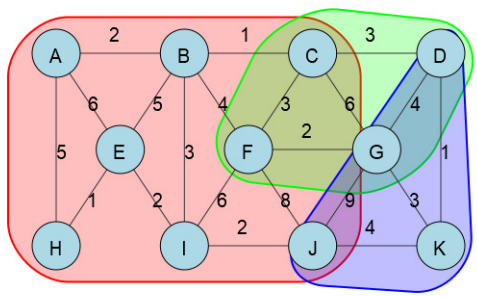

Fig. 4. Application of Algorithm 1B to Example 3.2 with gravitational weights.

Solution 2 (Detection of overlapping communities using Algorithm 4 in Example 3.2 with gravitational weights) The step-by-step application of the first and second stages of Algorithm 4 is shown in Table 4.

After applying the first 2 stages of Algorithm 4, we obtain 4 communities:

$$C_1 = \{A, B, C, E, F, H, I\}$$
$$C_2 = \{C, D, F, G, J, K\}$$
$$C_3 = \{B, E, H, I, J\}$$
$$C_4 = \{F, G\}.$$

After the third, merging stage of Algorithm 4, the following 2 communities are found:

$$C_1 = \{A, B, C, E, F, G, H, I, J\}$$
$$C_2 = \{C, D, F, G, J, K\}.$$

We show these communities in Figure 5.

Step	(u,v)	F (uv)	CNW (uv)	CNW (vu)	Communities
0					{A},{B},{C},{D},{E},{F},{G},{H},{I},{J},{K}
1	BC	20			{A},{D},{E},{F},{G},{H},{I},{J},{K},{B,C}
2	DK	9			{A},{E},{F},{G},{H},{I},{J},{B,C},{K,D}
3	EH	8			{A},{F},{G},{I},{J},{B,C},{K,D},{E,H}
4	FG	6.25			{A},{I},{J},{B,C},{K,D},{E,H},{F,G}
5	EI	4	4	4	{A},{J},{B,C},{K,D},{I,E,H},{F,G}
6	IJ	4	4	4	{A},{B,C},{K,D},{I,J,E,H},{F,G}
7	AB	3.75	3.75	3.75	{A,B,C},{K,D},{I,J,E,H},{F,G}
8	BI	2.22	3.02	2.22	{A,B,C},{K,D},{B,E,H,I,J},{F,G}
9	CF	2.22	2.78	3.79	{A,B,C,F},{K,D},{B,E,H,I,J},{F,G}
10	GK	1.67	2.60	1.67	{A,B,C,F},{K,D,G},{B,E,H,I,J},{F,G}
11	BF	1.56			{A,B,C,F},{K,D,G},{B,E,H,I,J},{F,G}
12	CD	1.33	1.89	1.33	{A,B,C,F},{C,K,D,G},{B,E,H,I,J},{F,G}
13	DG	0.94			{A,B,C,F},{C,K,D,G},{B,E,H,I,J},{F,G}
14	BE	0.80			{A,B,C,F},{C,K,D,G},{B,E,H,I,J},{F,G}
15	JK	0.75	1.00	0.75	{A,B,C,F},{C,D,G,J,K},{B,E,H,I,J},{F,G}
16	CG	0.56			{A,B,C,F},{C,D,G,J,K},{B,E,H,I,J},{F,G}
17	FI	0.56	2.43	2.78	{A,B,C,F,I},{C,D,G,J,K},{B,E,H,I,J},{F,G}
18	AE	0.33	4.32	5.13	{A,B,C,E,F,I},{C,D,G,J,K},{B,E,H,I,J},{F,G}
19	FJ	0.31	8.79	4.31	{A,B,C,E,F,I},{C,D,F,G,J,K},{B,E,H,I,J},{F,G}
20	GJ	0.25			{A,B,C,E,F,I},{C,D,F,G,J,K},{B,E,H,I,J},{F,G}
21	AH	0.24	4.32	8.24	{A,B,C,E,F,H,I},{C,D,F,G,J,K},{B,E,H,I,J},{F,G}

Tab. 4. Application of the first and the second stages of Algorithm 4 to Example 3.2 with gravitational weights.

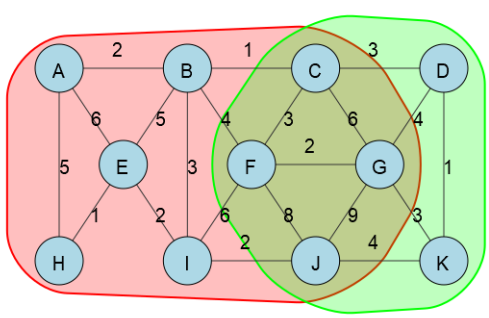

Fig. 5. Overlapping Communities after Applying Algorithm 4 to Example 3.2 with gravitational weights.

3.2. Proposed disjoint community detection algorithm

In this subsection, we explain our proposed disjoint community detection algorithm. The main idea of our proposed algorithm is as follows: First, we find overlapping communities with the help of Algorithm 4. Then, we consider the vertices that are in more than one community one by one. For example, let vertex u be in k communities, namely $u \in C_1$, $u \in C_2$,..., $u \in C_k$. We calculate the gravity force by which u is included in each community. For each i, we find the sum of the gravitational weights of the edges formed by the vertex u and the vertices in the community C_i. For whichever community this sum is the largest, we keep the vertex u in that community and delete it from all other communities. If there is more than one community where the maximum sum is obtained, we randomly choose one of these communities to hold the vertex u.

Let us explain the proposed algorithm on the Example 3.2. After applying Algorithm 4, we obtain two overlapping communities:

$$C_1 = \{A, B, C, E, F, G, H, I, J\}$$

$$C_2 = \{C, D, F, G, J, K\}.$$

Both communities have vertices C, F, G, J. The force holding the vertex C in the community C_1 is the sum $F(CB) + F(CF) + F(CG)$. The force holding this vertex in the community C_2 is the sum $F(CD) + F(CF) + F(CG)$. Since vertices F and G belong to both communities, it is sufficient to compare $F(CB)$ with $F(CD)$. As can be seen from Table 4, $F(CB) = 20 > F(CD) = 1.33$. Therefore, we delete vertex C from community C_2 and leave it in community C_1. The gravity force on the vertex F by the vertices in C_1 is equal to the sum $F(FB) + F(FC) + F(FG) + F(FI) + F(FJ)$. The gravity force on this vertex by the vertices in C_2 is equal to $F(FC) + F(FG) + F(FJ)$. This means that the vertex F is expelled from C_2. Proceeding similarly, we delete the vertex G from C_1, and the vertex J from C_2. As a result, we obtain the following disjoint communities:

$$C_1 = \{A, B, C, E, F, H, I, J\}$$

$$C_2 = \{D, G, K\}.$$

We show these disjoint communities in Figure 6.

To write the pseudocode of the proposed algorithm, let us denote by C_u the community to which a vertex u belongs. We define the gravitational force acting on the vertex u by the vertices in the community C_u as follows:

$$F(C_u) = \sum_{v \in C_u} F(uv). \tag{5}$$

The pseudo-code of the proposed algorithm to find disjoint communities is given in Algorithm 5.

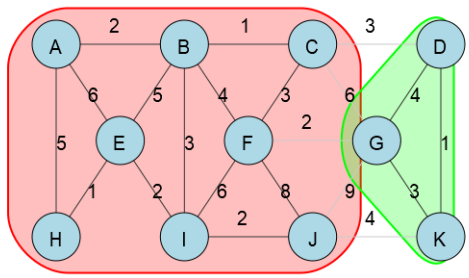

Fig. 6. Disjoint Communities after Applying Algorithm 5 to Example 3.2 with gravitational weights.

Algorithm 5: detectDisjointCommunities(G,w).

Input : $G = (V, E)$ is a weighted simple graph and $w(u, v)$ are the weights
Output : C is the list of disjoint communities

/* Stage 1: Finding of overlapping communities */
$C \leftarrow$ Algorithm 4(G, w)
/* Stage 2: Generating disjoint communities */
for *each vertex* $u \in V$ **do**
  // We count the number of communities that the vertex u is included in and assign it to the variable s.
  $s \leftarrow |\{C_u : C_u \in C\}|$
  if $s > 1$ **then**
    $F_u \leftarrow \max_{C_u \in C} F(C_u)$
    $C_u^* = arg\,max\ F_u$
    Delete vertex u from all C_u communities except C_u^*
  end
end
return C

4. EXPERIMENTAL RESULTS

We present the experimental evaluation of our proposed community detection algorithms on diverse network structures. Algorithm 4 refers to the overlapping community detection method, while Algorithm 5 corresponds to the disjoint variant.

Our experiments cover both synthetic and real-world networks (summarized in Table 5), focusing on three key aspects: structural quality, detection accuracy, and computational efficiency. The real-world datasets include Dolphins [57], Karate [58], Polbooks [59], Facebook [60], and Netscience [61], which vary in size and topology. Among these, the Netscience dataset is particularly notable for its weighted structure. As a co-authorship network, its edge weights represent tie strength based on the number of joint publications between authors [37]. This is particularly important because our algorithms directly utilize edge weights to model gravitational attraction between nodes, making weighted networks like Netscience especially suitable for revealing meaningful

Network Name	Number of Nodes	Number of Edges
Dolphins	62	159
Karate	34	78
Polbooks	105	441
Facebook	4039	88234
Netscience	1589	2742

Tab. 5. Network properties of well-known datasets that are used in the experiments.

community structures.

To further evaluate the robustness and generalization ability of our method, we conduct experiments on synthetic LFR benchmark networks generated using the parameter settings listed in Table 6. In these networks, N denotes the number of nodes, and E represents the total number of edges. The parameter k specifies the average node degree, while $maxk$ defines the maximum allowable degree. The edge weight mixing parameter muw determines the proportion of each node's strength that connects to nodes outside its community, thereby influencing inter-community connectivity. The parameter on defines the number of overlapping nodes, while om defines how many communities each overlapping node belongs to. For disjoint community detection experiments, we set $on = 0$ and $om = 0$, effectively eliminating node membership in multiple communities.

These parameter settings result in synthetic networks with clearly defined but overlapping community structures, allowing for rigorous evaluation of the algorithm's effectiveness under both moderate and large-scale conditions.

Network Name	N	E	k	maxk	muw	on	om
Network 1,2,3	500	2500	10	30	0.05	30	2
Network 4,5,6	1000	5000	10	30	0.1	30	2

Tab. 6. Network properties of LFR benchmark datasets used in overlapping community detection experiments.

4.1. Evaluation of overlapping community detection algorithm

Table 7 shows that our algorithm performs well on several real-world networks. Netscience achieves the highest Q_{oc} score of 0.90. This reflects the algorithm's ability to capture complex, weighted community structures. Similarly, the Facebook dataset exhibits a solid Q_{oc} score of 0.55, reflecting meaningful community organization in a large-scale social network.

Our algorithm also performs well on other metrics. Where ground-truth labels exist, NMI_{LFK}, Omega, and F-score show strong agreement. The Dolphins network reaches an F-score of 0.92, underscoring accurate community assignments. Collectively, these metrics highlight the robustness and reliability of our overlapping community detection algorithm across diverse network topologies and sizes. Note that for some datasets, such as Facebook and Netscience, ground-truth community labels are not available.

Network Name	Detected Communities	Runtime (s)	Q_{oc}	NMI_{LFK}	Omega	F-Score
Dolphins	2	0.001	0.31	0.58	0.66	0.92
Karate	2	0.001	0.11	0.29	0.15	0.69
Polbooks	2	0.003	0.43	0.51	0.63	0.68
Facebook	5	9.361	0.55	NA	NA	NA
Netscience	430	0.274	0.90	NA	NA	NA

Tab. 7. Results of the overlapping community detection algorithm on well-known datasets.

Therefore, evaluation metrics dependent on these labels (NMI, Omega, and F-Score) are marked as NA.

The results on synthetic overlapping networks are presented in Table 8. For smaller networks (Networks 1, 2, and 3), the algorithm shows strong alignment with ground-truth communities across multiple metrics, including high Q_{oc}, NMI, Omega index, and F-Score. This demonstrates its effectiveness in detecting overlapping structures within networks of moderate size and overlap. In larger, more complex networks (Networks 4, 5, and 6), Q_{oc} scores decline slightly. However, the algorithm remains accurate and stable. This shows good adaptability to increased complexity. Moreover, runtime increases moderately with network size but remains computationally efficient, supporting scalability.

Network Name	True Communities	Detected Communities	Runtime (s)	Q_{oc}	NMI_{LFK}	Omega	F-Score
Network 1	36	40	0.061	0.71	0.79	0.64	0.83
Network 2	33	40	0.062	0.71	0.76	0.71	0.84
Network 3	29	33	0.065	0.70	0.70	0.61	0.77
Network 4	71	125	0.272	0.64	0.62	0.57	0.83
Network 5	74	106	0.255	0.65	0.60	0.55	0.79
Network 6	66	101	0.250	0.66	0.62	0.55	0.79

Tab. 8. Results of the overlapping community detection algorithm on LFR benchmark datasets.

In summary, Algorithm 4 consistently agrees with ground-truth communities across various network sizes and overlapping configurations, confirming its robustness and scalability.

We emphasize again that our proposed algorithm works better in weighted graphs, where the meaning of the weights is the closeness of the vertices to each other. We can see this in Table 7 on the Netscience dataset. To apply the algorithm, all weights were set equal to 1 in unweighted graphs. We also tried the following: We applied the algorithm by assigning random weights to the edges in unweighted graphs as follows: Let the degree of the vertex with the highest degree in the graph be max_degree. Each edge is assigned a random integer in the range $[1, max\_degree+5]$. In this case, we found 30 overlapping communities on the Facebook set with the metric value $Q_{oc} = 0.66$.

In Table 9, we compare our proposed algorithm with the algorithms of Cetin and

Algorithm	Detected Communities	Q_{oc}
Cetin and Emrah Amrahov [10]	81	0.71
Chi et al. [11]	364	0.64
Proposed	430	0.90

Tab. 9. Comparison of the overlapping community detection algorithms on the Netscience dataset.

Emrah Amrahov [10] and Chi et al. [11] on the Netscience dataset. We chose the Netscience dataset for comparison since it is weighted complex network.

4.2. Evaluation of disjoint community detection algorithm

Disjoint community detection performance is evaluated on both real-world and synthetic networks. As shown in Table 10, the highest modularity score is observed in the Netscience network ($Q_{oc} = 0.87$). Smaller networks such as Karate and Dolphins exhibit lower modularity scores, likely due to simpler structures.

The Facebook network, being substantially larger, requires more runtime but still achieves moderate modularity, indicating identifiable community structure. The Polbooks and Dolphins networks also show moderate to high scores across evaluation metrics, while Karate's lower scores suggest less clearly defined communities.

Network Name	Detected Communities	Runtime (s)	Q_{oc}	NMI_{LFK}	Omega	F-Score
Dolphins	2	0.001	0.37	0.70	0.81	0.94
Karate	2	0.001	0.12	0.29	0.15	0.61
Polbooks	2	0.003	0.46	0.53	0.70	0.68
Facebook	5	9.698	0.55	NA	NA	NA
Netscience	430	0.291	0.87	NA	NA	NA

Tab. 10. Results of the disjoint community detection algorithm on well-known datasets.

The disjoint community detection results on synthetic LFR networks are summarized in Table 11. For smaller networks, the algorithm achieves high Q_{oc}, NMI, and F-Score values, indicating robust and accurate detection of well-separated communities. In larger networks, although modularity and other metrics show a slight decline, the algorithm maintains solid accuracy and consistent performance, demonstrating effective adaptation to increased network size and complexity.

Comparing these results with those of the overlapping community detection (Table 8), the disjoint algorithm generally performs better on networks with clearly separated, non-overlapping communities, reflected by higher Q_{oc} and F-Score in smaller networks. Conversely, the overlapping detection method excels in networks featuring significant node overlap, capturing more nuanced community structures despite sometimes achieving slightly lower Q_{oc}.

Our findings confirm that the proposed algorithms are robust and versatile. The

Network Name	True Communities	Detected Communities	Runtime (s)	Q_{oc}	NMI_{LFK}	Omega	F-Score
Network 1	29	32	0.111	0.86	0.89	0.93	0.97
Network 2	35	38	0.069	0.84	0.86	0.90	0.94
Network 3	28	34	0.067	0.76	0.78	0.85	0.91
Network 4	64	86	0.308	0.76	0.73	0.87	0.90
Network 5	73	92	0.276	0.74	0.73	0.82	0.88
Network 6	66	81	0.262	0.80	0.82	0.91	0.93

Tab. 11. Evaluation of disjoint communities in LFR benchmark networks.

disjoint method performs best on networks with clearly separated communities. This is evidenced by higher Q_{oc} and F-Score values in smaller, well-defined networks. Meanwhile, the overlapping method better captures complex community structures where nodes belong to multiple communities. It achieves higher scores in metrics sensitive to overlapping membership, such as NMI and Omega index, although sometimes with slightly lower Q_{oc} values. Both algorithms are computationally efficient and scale well across various network sizes and complexities. This makes them suitable for diverse applications depending on network characteristics.

All experiments were conducted on a system equipped with a 12th Gen Intel(R) Core(TM) i7-12700H CPU @ 2.30GHz, 16 GB RAM, running Windows 11 Home (64-bit).

5. CONCLUSION

In this study, we proposed two novel algorithms for detecting overlapping and disjoint communities in weighted complex networks. We model edge weights as distances between nodes and represent vertex similarity based on the universal law of gravitation, where each node's degree corresponds to its mass.

The overlap community detection algorithm uses a similarity-based approach adapted from a modified version of the Cetin and Emrah Amrahov method [10], forming communities by grouping nodes with strong mutual gravitational attraction. For disjoint community detection, nodes with multiple memberships are identified and assigned exclusively to the community exerting the strongest net gravitational pull, thereby removing them from other communities.

Extensive experiments on both real-world and synthetic networks validate the effectiveness and scalability of our methods. The algorithms consistently achieve high modularity and accuracy across diverse network topologies, demonstrating their suitability for complex, weighted networks with overlapping or disjoint community structures.

DATA AVAILABILITY

The code and data used in the experiments presented in this study are freely available on the following link:

`https://github.com/pelinercan89/GravitationalCommunityDetection`

ACKNOWLEDGEMENT

The authors sincerely thank the Editor-in-Chief Prof. Sergej Čelikovský and anonymous reviewers for their work, valuable comments, and suggestions.

(Received June 28, 2025)

REFERENCES

[1] R. Alguliyev, R. Aliguliyev, and L. Sukhostat: Method for quantitative risk assessment of cyber-physical systems based on vulnerability analysis. Kybernetika *60* (2024), 6, 779–796. DOI: 10.14736/kyb-2024-6-0779

[2] S. E. Amrahov, Y. Ar, B. Tugrul, B. E. Akay, and N. Kartli: A new approach to Mergesort algorithm: Divide smart and conquer. Future Gener. Computer Systems *157* (2024), 330–343. DOI:10.1016/j.future.2024.03.049

[3] S. E. Amrahov and B. Tugrul: A community detection algorithm on graph data. In: 2018 International Conference on Artificial Intelligence and Data Processing (IDAP) IEEE (2018), pp. 1–4. DOI:10.1109/IDAP.2018.8620850

[4] Y. Ar: An initialization method for the latent vectors in probabilistic matrix factorization for sparse datasets. Evol. Intell. *13* (2020), 2, 269–281. DOI:10.1007/s12065-019-00299-2

[5] M. Arasteh, S. Alizadeh, and C. G. Lee: Gravity algorithm for the community detection of large-scale network. J. Ambient Intell. Human Comput. *14* (2023), 1217—1228. DOI:10.1007/s12652-021-03374-8

[6] R. K. Bera and S. K. Mondal: Analyzing a transportation problem with reliability under the shadowed environment. Oper. Res. Forum *6* (2025), article number 69. DOI:10.1007/s43069-025-00469-2

[7] M. Bisht and S. Rawat: A novel interval-valued neutrosophic model to solve uncertain transportation problems. OPSEARCH (2025), 1–35. DOI:10.1007/s12597-025-00948-4

[8] P. Cetin and Ö. Tanriöver: Priority rule for resource constrained project planning problem with predetermined work package durations. J. Faculty of Engineering and Architecture of Gazi University *35* (2020), 3, 1537–1549. DOI:10.17341/gazimmfd.545873

[9] P. Cetin and S. E. Amrahov: A new network-based community detection algorithm for disjoint communities. Turkish J. Electr. Engrg. Comput. Sci. *30* (2022), 6, Article 13. DOI:10.55730/1300-0632.3933

[10] P. Cetin and S. Emrah Amrahov: A new overlapping community detection algorithm based on similarity of neighbors in complex networks. Kybernetika *58* (2022), 2, 277–300. DOI: 10.14736/kyb-2022-2-0277

[11] K. Chi, H. Qu, and Z. Fu: A novel approach for overlapping community detection in social networks based on the attraction. J. Comput. Sci. *85* (2025), 102508. DOI:10.1016/j.jocs.2024.102508

[12] S. Dhanasekar, J. J. Rani, and M. Annamalai: Transportation problem for interval-valued trapezoidal intuitionistic fuzzy numbers. Int. J. Fuzzy Logic Intell. Systems *22* (2022), 2, 155–168. DOI:10.5391/IJFIS.2022.22.2.155

[13] J. H. Chin and K. Ratnavelu: Community detection using constrained label propagation algorithm with nodes exemption. Computing *(*104) (2022), 339—358. DOI:10.1007/s00607-021-00966-2

[14] A. Ghareeb, O. Nooruldeen, C. A. Arslan, and J. K. Choi: Synergistic optimization of predictive models for water quality analysis in treatment plants using machine learning and evolutionary algorithms. Evolut. Intell. *18* (2025), 2, 1–24. DOI:10.1007/s12065-025-01022-0

[15] S. Goswami and A. K. Das: Determining maximum cliques for community detection in weighted sparse networks. Knowl. Inf. Syst. *64* (2022), 289—324. DOI:10.1007/s10115-021-01631-y

[16] H. Hajibabaei, V. Seydi, and A. Koochari: Community detection in weighted networks using probabilistic generative model. J. Intell. Inf. Syst. *60* (2023), 119—136. DOI:10.1007/s10844-022-00740-6

[17] E. Hazrati Nejad, S. Yigit-Sert, and S. Emrah Amrahov: An effective global path planning algorithm with teaching-learning-based optimization. Kybernetika *60* (2024), 3, 293–316. DOI: 10.14736/kyb-2024-3-0293

[18] Z. He, W. Chen, X. Wei, and Y. Liu: Mining statistically significant communities from weighted networks. IEEE Trans. Knowledge Data Engrg. *35* (2022), 6, 6073–6084. DOI: 10.1109/TKDE.2022.3176816

[19] N. Kartli: Hybrid algorithms for fixed charge transportation problem. Kybernetika *61* (2025), 2, 141–167. DOI: 10.14736/kyb-2025-2-0141

[20] N. Kartli, E. Bostanci, and M. S. Guzel: A new algorithm for optimal solution of fixed charge transportation problem. Kybernetika *59* (2023), 1, 45–63. DOI: 10.14736/kyb-2023-1-0045

[21] N. Kartli, E. Bostanci, and M. S. Guzel: Heuristic algorithm for an optimal solution of fully fuzzy transportation problem. Computing *106* (2024), 10, 3195–3227. DOI:10.1007/s00607-024-01319-5

[22] A. Khastan, B. H. Jimenez, and A. B. Moreno: On the new solution to interval linear fractional programming problems. Evolut. Intell. *17* (2024), 5, 4001–4005. DOI:10.1007/s12065-024-00968-x

[23] F. R. Khawaja, Z. Zhang, and A. Ullah: Common-neighbor based overlapping community detection in complex networks. Soc. Netw. Anal. Min. *15* (2025), Article number 61. DOI:10.1007/s13278-025-01480-5

[24] A. Kumar, P. Kumar, and R. Dohare: Revisiting neighbourhood proximity based algorithm for overlapping community detection in weighted networks. Soc. Netw. Anal. Min. *14* (2024), 105. DOI:10.1007/s13278-024-01257-2

[25] P. Kumar: A depth-first search approach to detect the community structure of weighted networks using the neighbourhood proximity measure. Int. J. Data Sci. Anal. (2024). DOI:10.1007/s41060-024-00631-9

[26] A. Lancichinetti, S. Fortunato, and J. Kertész: Detecting the overlapping and hierarchical community structure in complex networks. New J. Physics **11** (2009), 3, 033015. DOI:10.1088/1367-2630/11/3/033015

[27] S. Li, L. Jiang, X. Wu, W. Han, D. Zhao, and Z. Wang: A weighted network community detection algorithm based on deep learning. Appl. Math. Comput. *401* (2021), 126012. DOI:10.1016/j.amc.2021.126012

[28] W. Li, J. Wang, and J. Cai: New label propagation algorithms based on the law of universal gravitation for community detection. Physica A: Statistical Mechanics and its Applications *627* (2023) 129140. DOI:10.1016/j.physa.2023.129140

[29] H. Liu, Z. Li, and N. Wang: Overlapping community detection algorithm based on similarity of node relationship. Soft Comput *27* (2023), 19, 13689—13700. 10.1007/s00500-023-08067-2

[30] Z. Lu and Z. A. Dong: Gravitation-based hierarchical community detection algorithm for structuring supply chain network. Int. J. Comput. Intel. Syst. *16* (2023), 110. DOI:10.1007/s44196-023-00290-x

[31] J. Ma, L. Zhou, and J. Zuo: Adaptive community detection based on node dissimilarity. Int. J. Modern Physics C 2550066. DOI:10.1142/S0129183125500664

[32] A. McDaid, D. Greene, and N. Hurley: Normalized mutual information to evaluate overlapping community finding algorithms. arXiv preprint arXiv:1110.2515 (2011). DOI:10.48550/arXiv.1110.2515

[33] S. N. Mohammed and S. Gunduc: TPM: Transition Probability Matrix–Graph Structural Feature based Embedding. Kybernetika *59* (2023), 2, 234–253. DOI: 10.14736/kyb-2023-2-0234

[34] G. Murray, G. Carenini, and R. Ng: Using the Omega Index for evaluating abstractive community detection. In: Proc. Workshop on Evaluation Metrics and System Comparison for Automatic Summarization, Association for Computational Linguistics, 2012. pp. 10–18.

[35] K. Nallusamy and K. S. Easwarakumar: PERMDEC: community deception in weighted networks using permanence. Computing *106* (2024), 353—370. DOI:10.1007/s00607-023-01223-4

[36] E. Nasibov, M. Demir, and A. Vahaplar: A fuzzy logic apparel size decision methodology for online marketing. Int. J. Clothing Sci. Technol. *31* (2019), 2, 299–315. DOI:10.1108/IJCST-06-2018-0077

[37] M. E. J. Newman: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E, *74* (2006), 3, 036104. `https://doi.org/10.1103/PhysRevE.74.036104`

[38] B. Ozhan and B. Tugrul: Analysis of Turkish cuisine flavors network. Int. J. Food Sci. Technol. *59* (2024), 2, 908–915. DOI:10.1111/ijfs.16849

[39] S. Pandey and A. Gupta: Lazy merge sort: An improvement over merge sort. In: 2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT), Greater Noida 2024, pp. 1–6. DOI:10.1109/ICEECT61758.2024.10738877

[40] H S. Pattanayak, H. K. Verma, and A. L. Sangal: Gravitational community detection by predicting diameter. Discrete Math. Algorithms Appl. *14* (2022), 4, 2150145. DOI:10.1142/S1793830921501457

[41] P. Prokop, P. Drazdilova, and J. Platos: Overlapping community detection in weighted networks via hierarchical clustering. Plos one *19* (2024, 10, e0312596. DOI:10.1371/journal.pone.0312596

[42] G. Rossetti, L. Pappalardo, and S. Rinzivillo: A novel approach to evaluate community detection algorithms on ground truth. In: Complex Networks VII (2016), 133–144.

[43] S. Sandhiya and A. Dhanapal: Solving neutrosophic multi-dimensional fixed charge transportation problem. Contempor. Math. *5* (2024), 3, 3601–3624. DOI:10.37256/cm.5320244927

[44] L. Samandari Masooleh, J. E. Arbogast, W. D. Seider, U. Oktem, and M. Soroush: An efficient algorithm for community detection in complex weighted networks. AIChE J. *67* (2021), 7, e17205. DOI:10.1002/aic.17205

[45] M. Shen and Z. Ma: A novel node gravitation-based label propagation algorithm for community detection. Int. J. Modern Physics C *30* (2019), 6, 1950049. DOI: 10.1142/S0129183119500499

[46] J. Sheng, C. Liu, L. Chen, B. Wang, and J. Zhang: Research on community detection in complex networks based on internode attraction. Entropy *22* (2020), 12, 1383. DOI:10.3390/e22121383

[47] Shivani, D. Chauhan, and D. Rani: A feasibility restoration particle swarm optimizer with chaotic maps for two-stage fixed-charge transportation problems. Swarm Evolution. Comput. *91* (2024), 101776. DOI:10.1016/j.swevo.2024.101776.

[48] M. Subramaniam, T. Tripathi, and O. Chandraumakantham: Cluster Sort: A Novel Hybrid Approach to Efficient In-Place Sorting Using Data Clustering. IEEE Access *13* (2025), 74359–74374. DOI:10.1109/ACCESS.2025.3564380

[49] C. K. Suja, C. V. Harinarayanan, and A. Arivalagan: Novel objective-based coot puzzle optimisation for overlapping community expansion in complex networks. Int. J. Network. Virtual Organis. *31* (2024), 4, 281–305. DOI:10.1504/IJNVO.2024.144079

[50] Y. Wang, J. Chen, J. Bai, X. Lin, S. Liang, and Y. Zhang: Spatial-SLPA: uncovering overlapping communities in geospatial networks via the spatially constrained speaker-listener label propagation algorithm. In: International Conference on Smart Transportation and City Engineering (STCE 2024), SPIE, *13575* (2025), pp. 1247–1259). DOI:10.1117/12.3061956

[51] X. Wu, D. Teng, H. Zhang, J. Hu, Y. Quan, Q. Miao, and P. G. Sun: Graph reconstruction and attraction method for community detection. Appl. Intell. *55* (2025), 5, 1–17. DOI:10.1007/s10489-024-05858-4

[52] C. Yang, M. Li, and Y. Wang: Overlapping community detection algorithm based on the law of universal gravitation. In: MATEC Web of Conferences *22* (2015), 01056 EDP Sciences. DOI:10.1051/matecconf/20152201056

[53] H. B. Yıldırım, K. Kullu, and S. E. Amrahov: A graph model and a three-stage algorithm to aid the physically disabled with navigation. Univers. Access Inform. Soc. *23* (2024), 2, 901–911. DOI: 10.1007/s10209-023-00981-4

[54] Y. Y. Yu, C. Y. Xu, and K. F. Cao: An effective community detection method based on one-dimensional "attraction" in network science. Int. J. Modern Physics C *31* (2020), 5, 2050071. DOI:10.1142/S0129183120500710

[55] K. R. Žalik and B. Žalik: Node attraction-facilitated evolution algorithm for community detection in networks. Soft Comput. *23* (2019), 6135—6143. DOI:10.1007/s00500-018-3267-x

[56] H. Zhou, B. Xi, Y. Zhang, J. Li, and F. Zhang: A graph clustering algorithm using attraction-force similarity for community detection. IEEE Access *7* (2019), 13683–13692. DOI: 10.1109/ACCESS.2018.2889312

[57] Dolphins network dataset available at (accessed June 2025): github.com/csdashes/GraphStreamCommunityDetection/blob/master/data/dolphins.gml

[58] W. W. Zachary: Karate club network. Available via NetworkX documentation https://networkx.org/documentation/stable/reference/generated/networkx.generators.social.karate_club_graph.html, accessed June 2025.

[59] Political books network dataset available at (accessed June 2025): https://public.websites.umich.edu/∼mejn/netdata/

[60] Facebook social network dataset available at (accessed June 2025): http://snap.stanford.edu/data/

[61] Netscience co-authorship network dataset available at (accessed June 2025): https://public.websites.umich.edu/∼mejn/netdata/

Nermin Kartli, Ankara University, Department of Computer Science, Ankara, 06560. Turkey.
e-mail: nermin.kartli@gmail.com, nkartli@ankara.edu.tr

Pelin Çetin, Ankara University, Department of Computer Engineering, Ankara, 06830. Turkey.
e-mail: pelincetin.cs@gmail.com

Selin Ayhan, Ankara University, Department of Computer Engineering, Ankara, 06830. Turkey.
e-mail: selinayhan2022@hotmail.com