

Article

Overlapping Community Detection Based on Attribute Augmented Graph

Hanyang Lin ^{1,2}, Yongzhao Zhan ^{1,2,*}, Zizheng Zhao ³, Yuzhong Chen ³ and Chen Dong ³¹ School of Computer Science and Communications Engineering, Jiangsu University, Zhenjiang 212013, China; linyanhanson@163.com² Jiangsu Start Dima Data Processing Co., Ltd., Kunshan 215217, China³ College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China; zhzz7364@gmail.com (Z.Z.); yzchen@fzu.edu.cn (Y.C.); dongchen@fzu.edu.cn (C.D.)

* Correspondence: yzzhan@ujs.edu.cn

Abstract: There is a wealth of information in real-world social networks. In addition to the topology information, the vertices or edges of a social network often have attributes, with many of the overlapping vertices belonging to several communities simultaneously. It is challenging to fully utilize the additional attribute information to detect overlapping communities. In this paper, we first propose an overlapping community detection algorithm based on an augmented attribute graph. An improved weight adjustment strategy for attributes is embedded in the algorithm to help detect overlapping communities more accurately. Second, we enhance the algorithm to automatically determine the number of communities by a node-density-based fuzzy k-medoids process. Extensive experiments on both synthetic and real-world datasets demonstrate that the proposed algorithms can effectively detect overlapping communities with fewer parameters compared to the baseline methods.

Keywords: attributed networks; augmented attribute graph; community detection

Citation: Lin, H.; Zhan, Y.; Zhao, Z.; Chen, Y.; Chen, D. Overlapping Community Detection Based on Attribute Augmented Graph. *Entropy* **2021**, *23*, 680. <https://doi.org/10.3390/e23060680>

Academic Editor: Krzysztof Malarz

Received: 2 March 2021

Accepted: 23 May 2021

Published: 28 May 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Complex patterns exist in various real-world fields and can be simplified into complex networks. Individuals are represented as nodes, and the connections between them are correspondingly transformed to edges in a graph [1–3]. For example, the connections between proteins in organisms and the relationship between cities in a traffic system [4]—these real complex systems can be transformed into complex networks. Community detection is a common task in the field of complex network analysis [5,6].

Many studies have attempted to incorporate attribute and topology information in the community detection methodologies [7] beyond the traditional approaches [8,9]. In addition to the topological structure of nodes connected by edges, the nodes or edges themselves always carry attribute information—that is, they form an attributed network. The attributes can be used as complementary information to overcome the sparsity of topological structure [10,11]. However, these two sources of information may be contradictory to each other in some cases [12]. This makes it challenging to detect communities on attributed networks. In real-world networks, there may exist some vertices that belong to several communities simultaneously. Consequently, overlapping community detection has become a valuable research topic [13]. The proposed OCEA and AOCEA are implemented over attributed networks and can be used to detect overlapping communities. Most of the real-world networks are full of attribute information and overlapping communities are quite common in social networks. Therefore, the proposed algorithms will have a wide range of applications in real life.

The main contributions of this work are as follows: (1) We proposed two algorithms OCEA and AOCEA that can be used to detect overlapping communities on attributed

networks, which considers the multiple vertex-community memberships and employs a strategy to adjust attribute weights iteratively according to the memberships. (2) A new method for the automatic estimation of the number of communities is proposed to improve the practicability of OCEA. (3) Experimental results on synthetic and real-world datasets validate the effectiveness of OCEA and AOCEA.

Related work for Big Data Networks. Researches on community detection in large-scale networks are motivated by the situation that the traditional methods cannot handle the increasing size of networks. ADVNDS [14] utilized modularity maximization and designed a heuristic method to solve it. The algorithm proposed in [15] combined parallel processing techniques with binary trees to solve the efficiency problem. The algorithm proposed in [16] extended it later using CPU-GPU modules. CDMEC [17] introduced several functions to construct similarity matrices and integrated a stacked autoencoder and transfer learning to learn the embeddings of large-scale networks.

Related work on Attributed Networks. From the perspective of processing attribute information, existing algorithms can be classified into five categories [18]. First, algorithms based on distance design a distance function to combine attribute topology information. SToC [19] made use of the Jaccard index and Euclidean distance to measure the similarity of topology or attribute information, respectively. SA-cluster [20] obtains the distance by applying a random walk on an augmented attribute graph, and Inc-cluster [21] is a time-saving version of SA-cluster. Algorithms based on representation learning mainly focus on the process of learning the low-dimension vectors of nodes—that is, the embeddings. Community detection can be performed directly through clustering by embeddings. Potential information in the network can then be fully utilized by this method. MGAE [22] proposes a marginal graph convolutional network and obtained deeper representations through multiple autoencoders. DANE [23] employs two sub-processes to learn the attribute and topology representation. The final result is obtained by minimizing a designed negative log-likelihood. Evolutionary-algorithm-based methods measure the similarity of the topology and attribute information, and then transform the problem of community detection into a multi-objective optimization problem. BBO [24] proposes Simatt to represent the similarity of node attributes. Similarly, MOEA-SA [25] proposes SA to measure the attribute information. Nonnegative matrix factorization can also be used to obtain the representation of nodes. SCI [10] proposes a non-negative matrix factorization model with two sets of parameters. SCD [26] introduces an additional community relationship indicator matrix. The elements of the matrix describe the relationship between the corresponding communities. ASCD [27] introduces the concept of a mismatch between attributes and structural information, and then related adaptive parameters were added to detect communities. Finally, probabilistic generative model-based algorithms focus primarily on obtaining a generative model of communities. They directly transform the complex network structure into a probability model determined by several parameters. CESNA [28] models the interaction between a network structure and attributes to detect overlapping communities. The NEMBP [29] model utilizes nesting EM algorithms and confidence propagation to detect the communities based on the correlation between topology and attribute.

Related work on Overlapping Community Detection. The recent overlapping community detection algorithms can be classified into five categories. First, algorithms based on multi-objective evolutionary approach the global optimal solutions by swarm evolution [30–32]. MR-MOEA [31] introduces a mixed representation that consists of all the potential overlapping vertices and all the non-overlapping vertices. They evolve together to detect communities. MOGA-OCD [32] uses measures related to network connectivity to optimize two objectives: maximizing internal connectivity and minimizing external connectivity. Algorithms based on similarity partition vertices into communities according to their mutual similarity. OCDDP [33] proposes a method based on density peaks. LED [34] transforms the similarity into weights of the networks. Algorithms based on local expan-

sion first select initial vertices and then expand them to obtain communities. [35] optimizes the conductance community score to determine good seeds and then greedily expands them. [36] aims to find the structural centers of communities. Algorithms based on random walk utilize the path of random walk to define the connectivity among individuals. MCLC [37] employs a random walk on the edges and obtains “link communities” that are transformed into overlapping “node communities.” Finally, algorithms based on representation learning are similar to those based on attributed networks. Through underlying community membership, CDE [38] formulates community detection as a non-negative matrix factorization model based on the encoded community structures and attributes.

It is still challenging to detect overlapping communities on attribute networks. As introduced above in related work, only CESNA [28] and CDE [38] can solve the problem. To solve the problem, an overlapping community detection algorithm based on attribute augmented graph, OCEA, is proposed. We employ fuzzy k-medoids [39] on attribute augmented graph first proposed by [20] to obtain the communities. Furthermore, the number of communities k can be evaluated using the density of vertices. Through the evaluation, the automatic OCEA can detect overlapping communities without parameter k and obtain comparable or even better results comparing to the baseline methods.

The remainder of this paper is organized as follows. Section 2 introduces some preliminary information about the clustering problem and related definitions. Section 3 discusses the details of OCEA and its automatic variant. Section 4 presents empirical studies of the proposed algorithms. Finally, the conclusion and potential future work are reported in Section 5.

2. Preliminaries

2.1. Problem Definition

Attributed network can be denoted as $G = (V, E, A, X)$, where $V = \{v_1, v_2, \dots, v_n\}$ is a set with n vertices, $E = \{(v_i, v_j) | v_i, v_j \in V\}$ is a set of edges, $A = \{a_1, a_2, \dots, a_d\}$ denotes the set of attributes, and $X = (x_1, x_2, \dots, x_n)^T$ is an attribute matrix. Each row of X denotes the binary attribute vector of vertex i with d dimensions. If vertex i has attribute j , $a_j(v_i) = 1$. Otherwise, $a_j(v_i) = 0$. Figure 1 shows a traditional formulation of an attributed network. The attribute information of a vertex or an edge is regarded as a d -dimensional binary vector associated with the vertex or edge. In this study, we utilize a different formulation called the augmented attribute graph first proposed by [20] to detect overlapping communities on attributed networks.

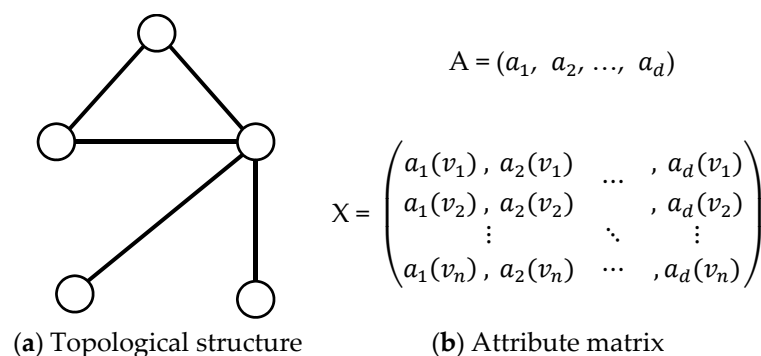


Figure 1. Illustration of an attributed network. For attribute matrix $X \in R^{n \times d}$, $X_{im} = 1$ when vertex i has the attribute m , otherwise $X_{im} = 0$.

The overlapping community detection problem on attributed networks can be discussed from two aspects:

- **Overlapping community:** In a traditional community detection problem, the final partitions do not share any vertices with each other. However, some vertices may be assigned to multiple communities in an overlapping community detection problem.

In this paper, we utilize the framework of fuzzy k-medoids [39] to calculate the memberships of every vertex to communities and finally obtain the overlapping communities.

- **Attributed network:** For an attributed network, the final partitions should satisfy two properties: (1) Topology similarity. The vertices belonging to the same community have more connections with each other than the vertices outside the community. (2) Attribute homogeneity. The vertices whose attribute vectors are close to each other have a high probability to be assigned to the same community. The community partitions should embody both the topology similarity and attribute homogeneity.

2.2. Augmented Attribute Graph

In contrast to the traditional formulation of an attributed network, the augmented attribute graph directly transforms the attributes into attribute vertices and adds them to the original graph. Figure 2 shows an example of an augmented attribute graph, the set of attributes $A = \{\text{Football}, \text{Basketball}\}$, and the set of structure vertices $V = \{v_1, v_2, \dots, v_5\}$. The solid points denote the added set of attribute vertices $V_a = \{v_{a1}, v_{a2}\}$, and the hollow points represent the original topology structure of the network. The solid lines denote the original structure edges. The hollow lines are used to connect the attribute and structure vertices. Vertex v_4 , for example, has two attributes: Basketball and Football. Thus, there are two attribute edges connected to the attribute vertices v_{a1} : Basketball and v_{a2} : Football. Consequently, the attribute vectors can be transformed into attribute vertices and edges on the augmented attribute graph.

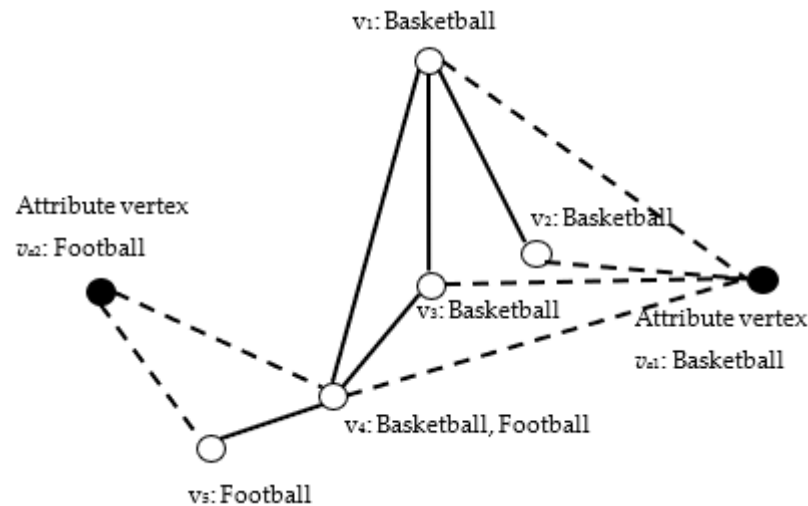


Figure 2. Example of augmented attribute graph.

Due to the addition of attribute vertices, the structure similarity and attribute homogeneity between vertices can be uniformly represented by the probability of random walks on the augmented attribute graph. Based on this, there are four cases to calculate the transition probability of random walk. The corresponding equations are expressed as follows:

1. From structure vertex v_i to structure vertex v_j :

$$p_{v_i, v_j} = \begin{cases} \frac{\omega_0}{N(v_i)\omega_0 + \omega_1 + \omega_2 + \dots + \omega_d}, & \text{if } (v_i, v_j) \in E \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

2. From attribute vertices v_{ip} and v_{jq} corresponding to the p th attribute of vertex v_i and the q th attribute of vertex v_j , respectively:

$$p_{v_{ip}, v_{jq}} = 0, \forall v_{ip}, v_{jq} \in V_a \quad (2)$$

3. From attribute vertex v_{ip} to structure vertex v_j :

$$p_{v_{ip}, v_j} = \begin{cases} \frac{1}{N(v_i)}, & \text{if } (v_{ip}, v_j) \in E_a \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

4. From structure vertex v_i to attribute vertex v_{jq} :

$$p_{v_i, v_{jq}} = \begin{cases} \frac{\omega_j}{N(v_i)\omega_0 + \omega_1 + \omega_2 + \dots + \omega_d}, & \text{if } (v_i, v_{jq}) \in E_a \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

where d is the dimension of attribute vectors. ω denotes the weights of the topology and attribute information, ω_0 denotes the weight of topology information, and $\omega_1, \omega_2, \dots, \omega_d$ denote the weights of attribute information. $N(v_i)$ denotes the number of neighbors of v_i on the original network. The indices p and q in Equations (2)–(4) could be equal or not, and $p, q \in \{a_1, a_2, \dots, a_d\}$.

Based on this, the transition probability of l step random walks between each pair of vertices can be obtained by Equation (5).

$$R_A = \sum_{\gamma=1}^l c(1-c)^\gamma P_A^\gamma \quad (5)$$

where c is the stop probability of the random walk, l is the step length and γ is a power. P_A is the transition probability matrix calculated by the Equations (1)–(4). Equation (5) denotes the process of l step random walks on the augmented attribute graph with a stop probability c . Here R_A is considered the similarity between vertices that reflects both topology and attribute information on the network.

3. Algorithms

Here, we introduce the proposed OCEA based on augmented attribute graph and its extended version with an estimation of the number of communities. The framework of OCEA is shown as Figure 3.

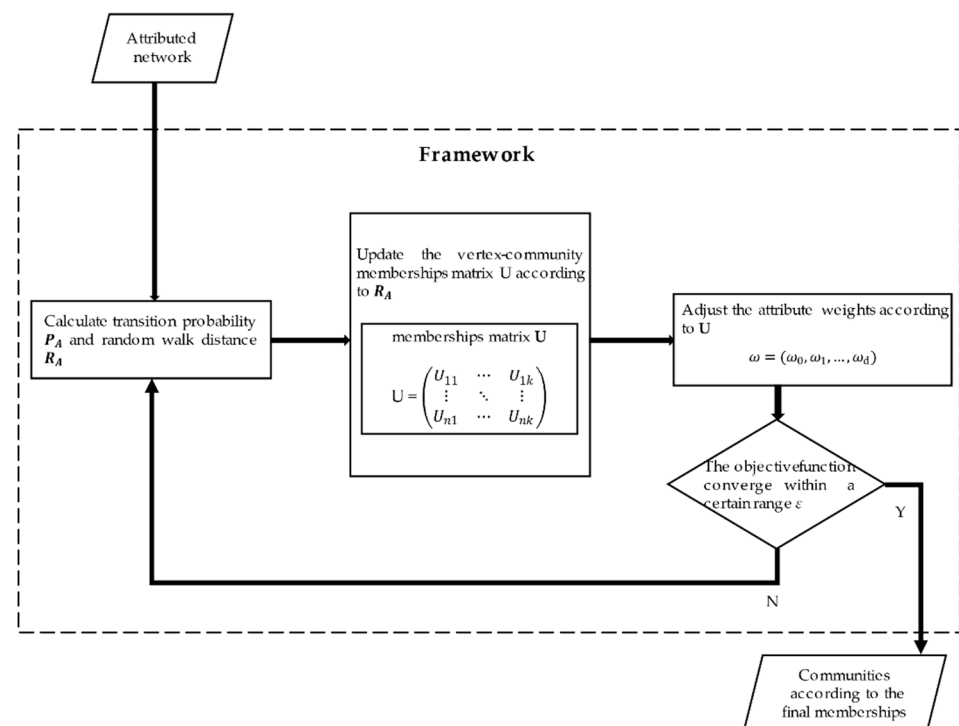


Figure 3. Framework of OCEA.

As shown in Figure 3, OCEA is mainly composed of three steps. First, it calculates the transition probability matrix P_A and the random walk similarity matrix R_A . Second, the vertex-community membership matrix U is updated according to R_A . Third, the structural weight ω_0 is fixed to 1, and the attribute weights $\omega_1, \omega_2, \dots, \omega_d$ are updated according to matrix U . The procedure is repeated until the objective function converges within a certain range ε .

3.1. Overlapping Community Detection

Based on the framework of the augmented attribute graph, the OCEA utilizes fuzzy k-medoids [39] to detect overlapping communities. i th row of the random walk similarity matrix R_A is used as the vector of v_i that is denoted as $R_A(v_i)$. Before the iteration of updating memberships, k initial vertices are selected as the centers of each cluster.

Equations (6) and (7) show the process of updating memberships matrix U and the vector X_c for centers in each cluster:

$$U_{ij}^{(t)} = \frac{1}{\sum_{c=1}^k \left(\frac{\|R_A^{(t)}(v_i) - X_c^{(t)}(j)\|}{\|R_A^{(t)}(v_i) - X_c^{(t)}(k)\|} \right)^2} \quad (6)$$

$$X_c^{(t+1)}(j) = \frac{\sum_{i=1}^N (U_{ij}^{(t)})^m \cdot R_A^{(t)}(v_i)}{\sum_{i=1}^N (U_{ij}^{(t)})^m} \quad (7)$$

where $U_{ij}^{(t)}$ denotes the memberships of vertex v_i to cluster j . Further, $X_c(j)$ is not the vector of an actual existing vertex that is regarded as the center of cluster j . It denotes the average vector of all vertices in cluster j . And $X_c(j)$ is used to update the memberships of all vertices in cluster j . Thereafter, the center vertices of each cluster are selected as follows:

$$c_i^{(t+1)} = \arg \min_{v_j \in V_i} \sum_{v_i \in V_i} \frac{\|R_A^{(t)}(v_j) - R_A^{(t)}(v_i)\|}{|V_i|} \quad (8)$$

The main objective of the algorithm is to ensure the vertices are close to their corresponding cluster centers. To this end, the objective function is minimized by

$$\min F = \sum_{i=1}^N \sum_{j=1}^k U_{ij}^m \|R_A(v_i) - R_A(c_j)\|^2 \quad (9)$$

where c_j denotes an actual existing vertex that is the center of cluster j determined by Equation (8). It is different from $X_c(j)$ in Equation (7). The m in Equation (7) and Equation (9) is a parameter and it will be set to 2 in experiments. The Euclidean distance is used to measure the similarity between vertices, after which the vertices are assigned to different communities according to their memberships to the communities.

3.2. Weights Adjustment

Each iteration of the update can obtain memberships. The weights of each attribute can be adjusted based on the currently detected communities. The equations of attribute weights adjustment are as follows:

$$\text{weight}_p(c_j, v_i) = \begin{cases} U_{ij}, & \text{if } c_j, v_i \text{ share the same value on } a_p \\ 0, & \end{cases} \quad (10)$$

$$\omega_i^{t+1} = \frac{1}{2} \left(\omega_i^t + \frac{d \sum_{j=1}^k \sum_{v \in V_j} \text{weight}_i(c_j, v)}{\sum_{p=1}^d \sum_{j=1}^k \sum_{v \in V_j} \text{weight}_p(c_j, v)} \right) \quad (11)$$

Memberships can be observed as the influence of a vertex on a cluster. The high similarity of a certain attribute in a cluster implies that the attribute is an effective feature to detect communities. Subsequently, the weights of the corresponding attributes should be increased, or else they should be decreased.

3.3. Estimation on the Number of Communities

In the case of community detection without ground truth, we cannot obtain the number of communities k directly. Most algorithms need to input k as a hyperparameter or directly set k as a fixed value. Based on OCEA, the automatic version of it is proposed using a process for estimating the number of communities k . It will be called AOCEA for brevity in the following.

First, the density of vertex is defined by Equation (12):

$$D(v_i) = \sum_{v_j \in V'} \left(1 - e^{-\frac{R_d(v_i, v_j)^2}{2}} \right) \quad (12)$$

Taking out four egonetworks randomly from the Facebook egonetwork set introduced in Section 4.1, we draw the histogram of the probability density distribution of each egonetwork's vertex density value in Figure 4. When calculating the density, the structural weight ω_0 is set to 1, and the attribute weights $\omega_1, \omega_2, \dots, \omega_d$ are set to $1/d$. The stop probability c is set to 0.5 to draw the curves.

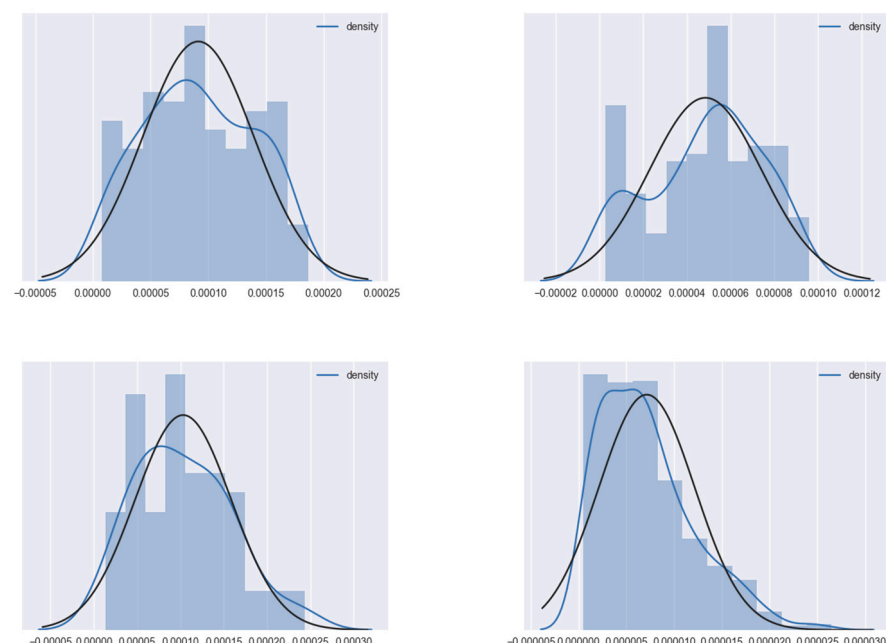


Figure 4. Probability distribution curve of four Facebook egonetworks' density values.

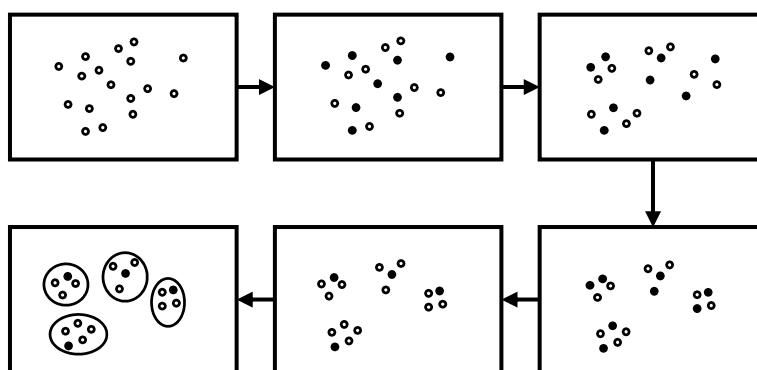
The x-axis denotes the vertex density value and the y-axis denotes the number of vertices. Each blue curve denotes the probability distribution curve fitted to each histogram. Each black curve denotes the Gaussian distribution calculated according to the mean and standard deviation of each subnetwork's vertex density value, as shown in Table 1.

Table 1. Statistics of the vertex density values of the Facebook’s egonetworks.

Network	Mean	Variance	Standard Deviation
Facebook_686	9.134×10^{-5}	2.294×10^{-9}	4.804×10^{-5}
Facebook_414	4.849×10^{-5}	6.579×10^{-10}	2.573×10^{-5}
Facebook_698	1.031×10^{-4}	3.079×10^{-9}	5.591×10^{-5}
Facebook_3437	7.135×10^{-6}	2.517×10^{-11}	5.021×10^{-6}

As we can observe in Figure 4, the fitted curves are very close to their corresponding calculated Gaussian distribution curves, which means that each sub-network’s vertex density value is nearly Gaussian distributed. It is assumed that centers should have higher density values than other vertices—higher density values imply a higher probability of being a center. Therefore, the sample mean of the density value is used as the threshold for selecting the candidate initial centers.

As shown in Figure 5, the solid points denote the initial selected candidate centers based on the density value. In general, the number of initial centers was much larger than the number of real communities. Under the framework of OCEA, with the iterative update of centers and attribute weights using Equations (8) and (11), respectively, the l step random walk probability changes and influences the similarity between vertices. In general, the vertices in the network gradually gather to several clusters in a distinct trend because of the adjustment of weights.

**Figure 5.** Estimation on the number of communities according to the evolution of community centers.

During the update of memberships, there may be several clusters that select the same vertex as their centers—that is, there may exist a vertex whose memberships to several clusters are all the highest in the current cluster. Only the vertices with the highest memberships are preserved as centers. This implies that parts of the candidate centers will be eliminated. As shown in Figure 5, parts of the solid points are transformed into hollow points, which implies that the solid points are no longer regarded as centers. After the elimination of centers, the number of remaining centers can be observed as an approximation of the number of communities. Function 1 summaries the process of cluster number estimation.

Function 1. ClusterNumberEstimation.

Input: Attribute Augmented Graph $G_A(V, E, A, X)$, random walk length l , stop probability c , number of epochs T

Output: Number of Communities k

- 1: Initialize weights $\omega_0 = \omega_1 = \dots = \omega_d = 0$, $d_s = 0$
 - 2: **for** vertices $v_i \in V$ **do**
 - 3: Compute density $D(v_i)$ by Equation (12)
 - 4: $d_s = d_s + D(v_i)$
 - 5: **end for**
 - 6: $d_T = d_s/n$
 - 7: Initialize centers $V_D \leftarrow \{v_i | D(v_i) \geq d_T, v_i \in V\}$
-

```

8:   $L_k = |V_D|$ 
9:  for  $i = 1, \dots, T$  do
10:    for vertices  $v_i \in V$  do
11:      Update memberships  $U_{ij}$  and  $X_c$  by Equations (6) and (7)
12:    end for
13:    for  $m = 1, \dots, d$  do
14:      Update weights of attributes  $\omega_m$  by Equation (11)
15:    end for
16:    Compute  $R_A$  by Equation (5)
17:  end for
18:  for  $j = 1, \dots, L_k$  do
19:     $V_c \leftarrow \operatorname{argmax}_{v_i}(U_{ij})$ 
20:  end for
21:   $k = |V_c|$ 
22: return  $k$ 

```

3.4. OCEA and AOCEA

As introduced in the previous sections, the update of memberships and the adjustment of weights are the main processes of OCEA, as shown in Algorithm 1.

It initializes weights, the random walk similarity, and centers first (lines 1–3). Then, the memberships, centers, and weights of attributes are updated iteratively until the convergence of the objective function in a certain range (lines 4–17). Detected communities are obtained using memberships (lines 18–23).

Algorithm 1 OCEA

Input: Attribute Augmented Graph $G_A(V, E, A, X)$, random walk length l , stop probability c , convergence parameter ε , overlapping parameter γ , number of communities k

Output: Communities $C_1, \dots, C_k$

```

1:  Initialize weights  $\omega_0 = \omega_1 = \dots = \omega_d = 1$ 
2:  Compute initial  $R_A$  by Equation (5)
3:  Select the initial centers  $V_c$ 
4:  while  $mar > \varepsilon$  do
5:    for vertices  $v_i \in V$  do
6:      for  $j = 1, \dots, k$  do
7:        Update memberships  $U_{ij}$  by Equation (6)
8:        Update centers  $X_c$  by Equation (7)
9:      end for
10:    end for
11:    for  $m = 1, \dots, d$  do
12:      Update weights of attributes  $\omega_m$  by Equation (11)
13:    end for
14:    Compute  $R_A$  by Equation (5)
15:    Compute objective function  $F^t$ 
16:     $mar = |F^t - F^{t-1}|$ 
17:     $F^{t-1} = F^t$ 
18:  end while
19:  for vertices  $v_i \in V$  do
20:     $\mu_{max}^i = \max(\{U_{ij} \mid j = 1, \dots, k\})$ 
21:    for  $j = 1, \dots, k$  do
22:       $C_j \leftarrow \{v_i \mid U_{ij} \geq \gamma \times \mu_{max}^i\}$ 
23:    end for
24:  end for
25: return  $C_1, \dots, C_k$ 

```

As shown in Algorithm 2. AOCEA first utilizes the function *ClusterNumberEstimation* to obtain an estimated number of communities k . Then, this estimated k is regarded as the input of the OCEA to detect the overlapping communities.

Algorithm 2 AOCEA

Input: Attribute Augmented Graph $G_A(V, E, A, F)$, random walk length l , stop probability c , convergence parameter ε , overlapping parameter γ

Output: Community $C_1, \dots, C_k$

1: Initialize $k = \text{ClusterNumberEstimation}(G_A, l, c, T)$

2: return OCEA($G_A, l, c, \varepsilon, \gamma, k$)

The time complexity of computing the transition probability is $O(|E_A|)$. Here, E_A denotes the set of edges in an attribute augmented graph that consists of two parts. The edges in an attribute augmented graph composed of two parts. One includes the original edges in the topological graph, the other includes the added edges according to the attribute information. Let z_p be the numbers of nonzero entries in transition probability matrix P and n be the number of vertices. Since P is a sparse matrix, the time complexity of the matrix multiplication is $O(z_p^2 n)$. Weight adjustment requires $O(nk)$ time. Here, k is the true number of communities. Thus, the overall complexity of OCEA and AOCEA is $O(|E_A| + z_p^2 n + nk)$, which can be reduced to $O(n)$ because $k \ll n$. AOCEA has an extra procedure of estimating the number of communities, and its initial estimation of the number of communities is roughly equal to $n/2$. Therefore, the time complexity of weights adjustment in AOCEA is $O(n^2)$ and gradually reduces to $O(nk_e)$, where k_e is the final estimation of the number of communities.

4. Experiments

4.1. Datasets

The algorithms were tested on synthetic and real-world attributed networks.

We evaluated the performance of proposed algorithms and the baselines on six real-world attributed networks. The Facebook egonetwork set (<http://snap.stanford.edu/data/>) (accessed on 28 April 2021) is a set of ten Facebook users' ego networks. The users are denoted by nodeId's and each ego network is numbered by the specific nodeId. Vertices denote the users of Facebook and each two pair of vertices within a circle (friends lists) are connected by an edge with a certain probability. The user profiles are encoded into vectors with binary values [40].

Texas, Washington and Wisconsin are three citation networks, where the vertices denote papers and the edges denote the citation relationships among the papers. The features of the papers are encoded into binary values. The papers are classified according to their domains [41].

Synthetic networks and the true communities were generated by the LFR benchmark [42]. The methods in [43] were used to generate each network's attribute matrix according to its true communities. Specifically, the vertices in a community shared the same attributes with high probability while the probability for the vertices in different communities is low. The generated attributes were consistent with the format of .feat files of the real-world network Facebook.

Table 2 presents the meaning of the parameters for the LFR benchmark networks. Two sets of synthetic networks D_1 and D_2 were generated to evaluate the algorithms. D_1 is a set of seven networks with increasing community size from 100 to 700. D_2 is a set of five networks with increasing mixing parameters μ . The number of attributes was set to be the same as the number of communities, and τ_1 and τ_2 were set as default values 2.0 and 1.0, respectively. Table 3 lists the corresponding parameters of D_1 and D_2 .

Table 2. The meaning of parameters in LFR benchmark networks.

Parameter	Meaning
N	number of vertices
k_{avg}	average degree
k_{max}	maximum degree
μ	mixing parameters
τ_1	negative exponent of degree
τ_2	negative exponent of community size
c_{min}	minimum community size
c_{max}	maximum community size
on	number of overlapping vertices
om	maximum of an overlapping vertex belongs to

Table 3. Parameter settings of D₁ and D₂ datasets.

Dataset	Parameters
D ₁	$N = 100\text{--}700$, $k_{avg} = 10$, $k_{max}=25$, $\mu = 0.1$, $c_{min} = 10$, $c_{max} = 50$, $on = 20$, $om = 2$
D ₂	$N = 600$, $k_{avg} = 10$, $k_{max} = 25$, $\mu = 0.2\text{--}0.6$, $c_{min} = 10$, $c_{max} = 50$, $on = 20$, $om = 2$

In addition to synthetic networks, three egonet networks from Facebook egonet network set were also used in our experiments. As shown in Table 4, N is the number of vertices, M is the number of attributes, and k denotes the number of real communities.

Table 4. Information of real-world networks.

Network	N	M	k
Facebook_698	65	48	13
Facebook_348	225	161	14
Facebook_0	347	2533	24
Texas	187	328	5
Washington	230	446	5
Wisconsin	265	530	5

4.2. Evaluation Metric

Overlapping normalized mutual information (ONMI) [44] is an improved measure of normalized mutual information (NMI) that can be used as the evaluation metric for overlapping community detection. This metric reflects the accuracy of the detected communities. The higher the value of ONMI, the higher the similarity between detected communities and ground-truth communities. Unlike NMI, ONMI can measure the accuracy of overlapping communities.

Given two covers, detected communities X and ground truth Y , $H^*(X_i | Y_j)$ is defined as

$$H^*(X_i | Y_j) = \begin{cases} H(X_i | Y_j), & \text{if } h(a, n) + h(d, n) \geq h(b, n) + h(c, n) \\ h(c + d, n) + h(a + b, n), & \text{otherwise} \end{cases} \quad (13)$$

where $a = \sum_{m=1}^n [X_{im}=0 \wedge Y_{jm}=0]$, $b = \sum_{m=1}^n [X_{im}=0 \wedge Y_{jm}=1]$, $c = \sum_{m=1}^n [X_{im}=1 \wedge Y_{jm}=0]$, $d = \sum_{m=1}^n [X_{im}=1 \wedge Y_{jm}=1]$ and $h(w, n) = \log_2(w/n)$.

In Equation (13), i and j denote the index of the clusters. X_{im} and Y_{jm} denote whether there is a vertex m in the cluster i or j , respectively. Subsequently, the entropy $H(X | Y)$ and $H(X)$ can be obtained as

$$H(X | Y) = \sum_{i \in \{1, \dots, K_X\}} \min_{j \in \{1, \dots, K_Y\}} H^*(X_i | Y_j) \quad (14)$$

$$H(X) = \sum_{i=1}^{K_X} \left(h \left(\sum_{m=1}^n [X_{im} = 1], n \right) + h \left(\sum_{m=1}^n [X_{im} = 0], n \right) \right) \quad (15)$$

Next, $H(Y|X)$ and $H(Y)$ can be obtained similarly. ONMI is defined as

$$\text{ONMI} = \frac{\frac{1}{2} [H(X) - H(X|Y) + H(Y) - H(Y|X)]}{\max(H(X), H(Y))} \quad (16)$$

4.3. Experimental Scheme

Four experiments were conducted: runtime, the accuracy of community detection on synthetic or real-world networks, and the accuracy of the estimation of the number of communities.

The baseline methods include Inc-cluster, S-cluster, W-cluster [14], OCEA, and AOCEA. Inc-cluster utilizes random walk similarity as the similarity between each pair of vertices and k-means [45] to detect communities without considering overlapping communities. Its adjustment of weights does not consider the memberships of each vertex to communities.

S-cluster only considers a network's topology information, setting the structural weight ω_0 to 1.0 and other weights to 0.0. W-cluster is an algorithm that considers attribute information without weight adjustment. It sets the structural weight to 1.0 and the weight of each attribute to $1/d$. S-cluster and W-cluster can be regarded as the variants of Inc-cluster without adjustment of weights. The step length l is set to 5 for all algorithms. The power m in Equations (7) and (9) is set to 2. The ε is set to 10^{-5} . The stop probability c is varying from 0 to 1:

1. Accuracy of detected communities on synthetic networks: Experiments were conducted on D_1 and D_2 to compare the accuracy of detected communities. D_1 fixes the mixing parameter μ to 0.1 and sets the size of networks from 100 to 700. Meanwhile, D_2 fixes the size of the networks to 600 and sets the mixing parameters μ from 0.2 to 0.6.
2. Accuracy of detected communities on real-world networks: Experiments were conducted on three networks from Facebook egonetwork set and three paper citation networks: Texas, Washington, and Wisconsin. The accuracy of the detected communities was studied and analyzed.
3. Accuracy of estimation on the number of communities: Experiments on Facebook egonetwork set, including all the ten egonetworks, to study the accuracy of the estimation on real-world networks.
4. Runtime: Experiments were conducted on D_1 to compare the algorithm runtimes. As the size of networks increasing, we analyze the changing runtime trends.

The above experiments were conducted on a computer with an Intel i5-6300HQ CPU @ 2.30 GHz, with 8 GB RAM.

4.4. Results and Analysis

4.4.1. Accuracy on Synthetic Networks

The results of ONMI values on synthetic datasets D_1 and D_2 are shown in Figures 6 and 7.

Figure 6 shows that the ONMI value of algorithms is maintained within a certain range with an increasing number of vertices. OCEA had a higher ONMI value on D_1 than the other algorithms because they consider the vertex's memberships. S-cluster detects communities without considering attribute information, and W-cluster does not have the process of attribute weight adjustment. Consequently, the superiority of OCEA and AOCEA reflects the effectiveness of considering attribute information and corresponding

adjustment of weights. Besides, the adjustment of weights in OCEA and AOCEA considers the memberships of each vertex to communities. That is, the attribute weights of the vertices in communities are updated according to the memberships. Inc-cluster does not consider the attribute weights to communities in the adjustment of weights.

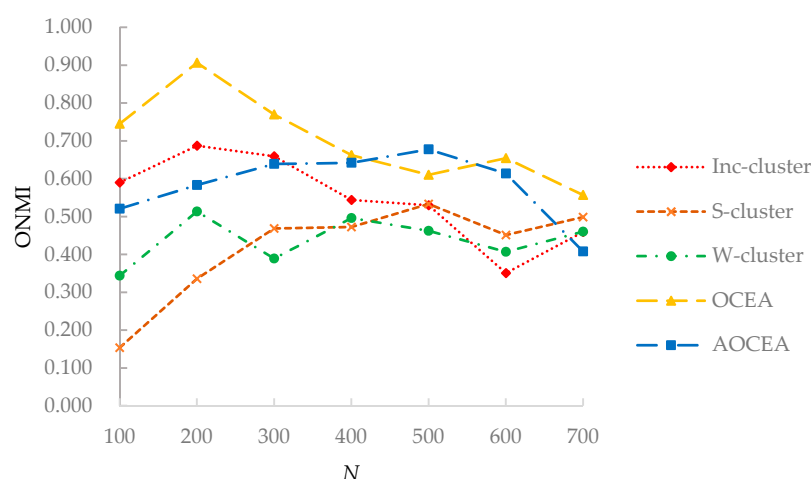


Figure 6. ONMI values of algorithms with varying network sizes.

As shown in Figure 7, with the increase of the values of the mixing parameter μ , it becomes difficult for the algorithms to uncover the communities with increasingly blurred boundaries, which results in the decrease of the ONMI values. All algorithms could not detect communities correctly when the value of μ reaches 0.6. Although Inc-cluster [21] exhibited a high ONMI value at the beginning, it was greatly affected by the increase of the value of μ . S-cluster and W-cluster were affected less than other algorithms because of their inadequate use of attribute information. OCEA had a similar changing trend of the ONMI value compared to Inc-cluster because of the introduction of attribute information. The ONMI value of AOCEA decreased greatly when $\mu > 0.3$ because it included an extra procedure of estimating the number of communities. The ONMI value of AOCEA increased when $\mu < 0.3$ because the estimated number of communities is affected by the update of memberships matrix U . And the boundaries between communities are not blurred when $\mu < 0.3$. The number of communities estimated by AOCEA may not be identical to the true ones when the value of μ is higher than 0.5, which affects its accuracy.

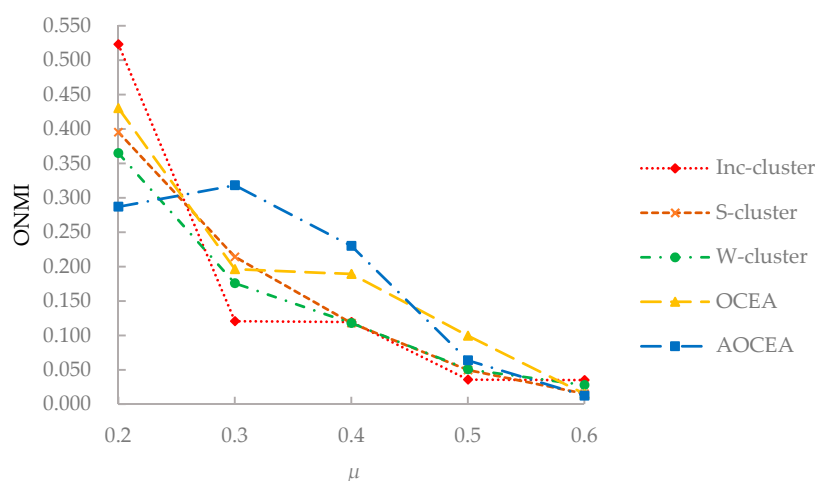


Figure 7. ONMI values of algorithms with varying values of μ .

4.4.2. Accuracy on real-world Networks

Six networks were selected to study the accuracy of algorithms on real-world networks. The results are shown in Figure 8.

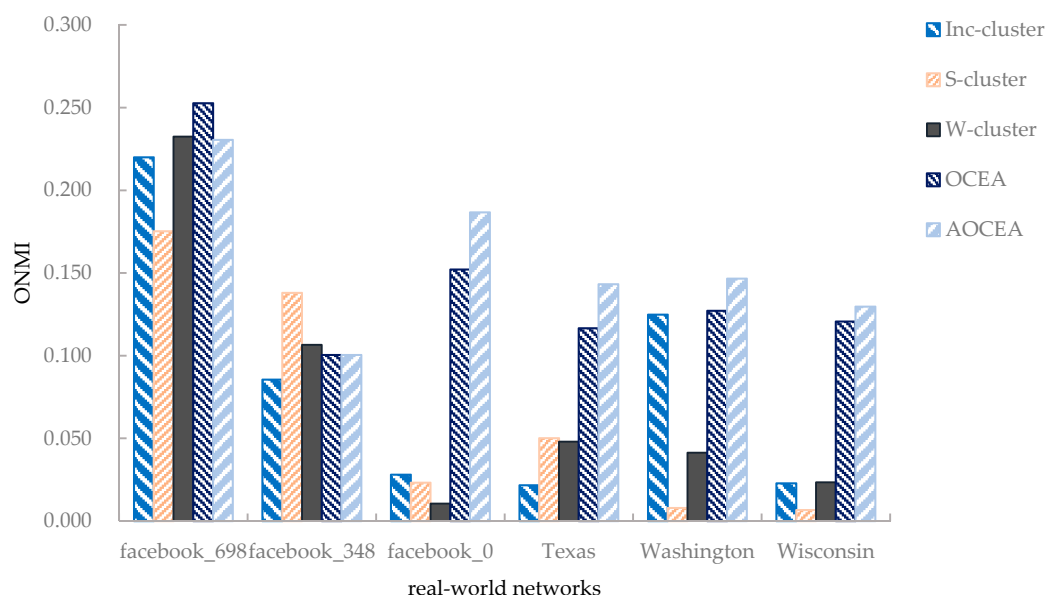


Figure 8. ONMI values of algorithms on real-world networks.

As shown in Figure 8, the accuracy of OCEA was relatively higher than that of Inc-cluster algorithm [21] on the Facebook egonetworks. Although S-cluster only considers topology information, it still performs well on some networks. It is assumed that there may have been some attribute information that was not conducive to detecting communities—that is, the introduction of some attributes may have degraded the results of community detection. Moreover, the invalid attribute information may have influenced the estimation of the number of communities, resulting in the inferior performance of AOCEA on some networks. OCEA and AOCEA were superior to all the baselines because they consider the vertices' community memberships and employ a strategy to adjust attribute weights iteratively according to the memberships. The vertices in the communities in the three paper citation networks (Texas, Washington and Wisconsin) are loosely connected, which results in the poor performance of S-cluster and W-cluster. Inc-cluster performed better than S-cluster and W-cluster because it considers the adjustment of attribute weights. However, it is not competitive with OCEA and AOCEA because it does not consider the vertices' community memberships.

On the whole, as we can observe from Figure 6, network size has little impact on the algorithms' accuracy when it is greater than 400. We conjecture that the accuracy of an algorithm on a synthetic network should exhibit little change when all network parameters in Table 2 except N are kept invariant because the more important structure and attribute features of the network are invariant. Additionally, as shown in Figure 8, the accuracy of an algorithm on a large network (facebook_0 with 347 vertices) may be inferior to that on a small network (facebook_698 with 65 vertices), which also reveals that other features of a network than its size may have a greater impact on its accuracy.

4.4.3. Accuracy of estimation on the number of communities

Experiments were conducted on all ten egonetworks from Facebook egonetwork set, and the number of vertices on each network was 347, 1045, 792, 755, 547, 227, 59, 159, 170, and 66, respectively. The results of the estimation of the number of communities are

shown in Table 5. Here, real, initial, and estimation denote the real number of communities, the number of selected initial candidate centers, and the estimated number of communities by AOCEA, respectively.

Table 5. Estimation on the number of communities in the Facebook’s egonetworks.

Facebook	Real	Initial	Estimation
facebook_0	24	127	17
facebook_107	9	417	17
facebook_1684	17	335	14
facebook_1912	46	338	12
facebook_3437	32	230	15
facebook_348	14	109	14
facebook_3980	17	27	14
facebook_414	7	85	19
facebook_686	14	76	8
facebook_698	13	28	16

As presented in Table 5, the number of initial candidate centers was much larger than the real ones. After the estimation, most of the estimated number of communities were generally close to the real ones except for facebook_1912, facebook_3437, and facebook_414. The gaps between the estimated and real number of communities on some networks were assumed to be caused by the sparse structure of communities on these networks.

4.4.4. Runtime

As shown in Figure 9, each algorithm exhibited a different trend with the increasing number of vertices. OCEA ran faster than Inc-cluster because the adjustment of attribute weights and the update of the centers in OCEA are more efficient than that in Inc-cluster. S-cluster and W-cluster methods do not contain the process of updating weights, leading to their fast running speed. The runtime of AOCEA is much higher than that of other algorithms because it contains an additional process to estimate the number of communities.

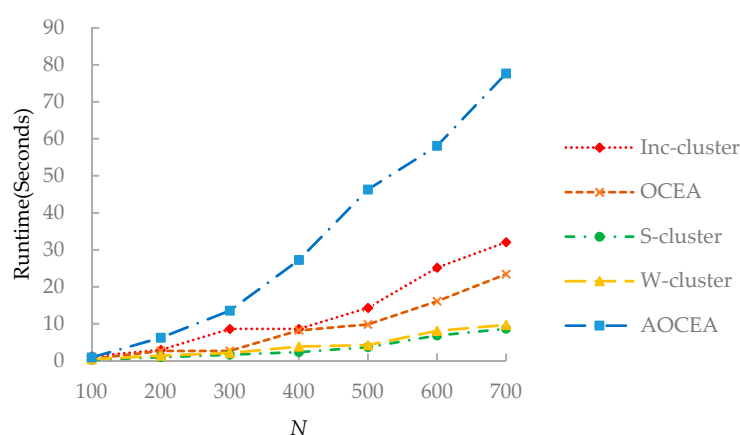


Figure 9. Runtime of the algorithms.

5. Conclusions

In this paper, an overlapping community detection algorithm called OCEA—based on an augmented attribute graph—and its extended version with an additional estimation process for the number of communities was proposed. To utilize the attribute information properly, we adopt a strategy of iteratively adjusting attribute weights. For overlapping

community detection, fuzzy k-medoids [39] was employed. In addition, a process to estimate the number of communities was introduced to solve the community detection problems despite the number of communities being unknown. Experimental results showed the effectiveness of the proposed algorithms in real-world and synthetic networks. In the future, we will continue to study the strategy to estimate the number of communities more precisely to increase the performance of AOCEA. We will also explore new heuristic methods to conduct biased random walk towards vertices with high attribute weights to compensate for the inaccuracy caused by the loss of structural information in community detection on sparse networks. Furthermore, we plan to adopt approximate random walk to improve our algorithms' performance on large networks.

Author Contributions: Conceptualization, H.L.; Methodology, Y.Z. and C.D.; Software, H.L.; Validation, H.L., Y.Z. and Z.Z.; Formal Analysis, H.L. and Z.Z.; Investigation, Y.Z. and Z.Z.; Resources, H.L.; Data Curation, Z.Z. and Y.C.; Writing—Original Draft Preparation, Z.Z. and Y.C.; Writing—Review & Editing, H.L., Y.Z. and C.D.; Visualization, Y.C.; Supervision, H.L.; Project Administration, Y.Z.; Funding Acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China under Grant No. 61672159, No. 61672158, No. 61300104 and No. 62002063.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in <http://snap.stanford.edu/data/>, accessed on 28 April 2021.

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

References

1. Albert, R.; Barabási, A.-L. Statistical mechanics of complex networks. *Rev. Mod. Phys.* **2002**, *74*, 47–97.
2. Fortunato, S. Community detection in graphs. *Phys. Rep.* **2010**, *486*, 75–174.
3. Zachary, W.W. An Information Flow Model for Conflict and Fission in Small Groups. *J. Anthropol. Res.* **1977**, *33*, 452–473.
4. Wang, F.Y.; Zeng, D.; Carley, K.M.; Mao, W. Social computing: From social informatics to social intelligence. *IEEE Intell. Syst.* **2007**, *22*, 79–83.
5. Lancichinetti, A.; Fortunato, S. Community detection algorithms: A comparative analysis. *Phys. Rev. E* **2009**, *80*, 056117.
6. Coscia, M.; Giannotti, F.; Pedreschi, D. A classification for community discovery methods in complex networks. *Stat. Anal. Data Min.* **2011**, *4*, 512–546.
7. Bothorel, C.; Cruz, J.D.; Magnani, M.; Mícenkova, B. Clustering attributed graphs: Models, measures and methods. *Netw. Sci.* **2015**, *3*, 408–444.
8. Xiao, J.; Ren, H.F.; Xu, X.K. Constructing Real-Life Benchmarks for Community Detection by Rewiring Edges. *Complexity* **2020**, *2020*, 7096230, doi:10.1155/2020/7096230.
9. Hoffmann, T.; Peel, L.; Lambiotte, R.; Jones, N.S. Community detection in networks without observing edges. *Sci. Adv.* **2020**, *6*, eaav1478, doi:10.1126/sciadv.aav1478.
10. Hou, C.; He, S.; Tang, K. RoSANE: Robust and scalable attributed network embedding for sparse networks. *Neurocomputing* **2020**, *409*, 231–243.
11. Sheikh, N.; Kefato, Z.; Montresor, A. gat2vec: Representation learning for attributed graphs. *Computing* **2019**, *101*, 187–209.
12. Wang, X.; Jin, D.; Cao, X.; Yang, L.; Zhang, W. Semantic community identification in large attribute networks. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, Phoenix, AZ, USA, 12–17 February 2016; Volume 2016, pp. 265–271.
13. Jin, D.; Li, B.; Jiao, P.; He, D.; Shan, H.; Zhang, W. Modeling with node popularities for autonomous overlapping community detection. *ACM Trans. Intell. Syst. Technol. (TIST)* **2020**, *11*, 1–23.
14. Džamić, D.; Aloise, D.; Mladenović, N. Ascent–descent variable neighborhood decomposition search for community detection by modularity maximization. *Ann. Oper. Res.* **2019**, *272*, 273–287.
15. Souravlas, S.; Sifaleras, A.; Katsavounis, S. A parallel algorithm for community detection in social networks, based on path analysis and threaded binary trees. *IEEE Access* **2019**, *7*, 20499–20519.
16. Souravlas, S.; Sifaleras, A.; Katsavounis, S. Hybrid CPU-GPU Community Detection in Weighted Networks. *IEEE Access* **2020**, *8*, 57527–57551, doi:10.1109/ACCESS.2020.2982227.
17. Xu, R.; Che, Y.; Wang, X.; Hu, J.; Xie, Y. Stacked auto-encoded based community detection method via an ensemble clustering framework. *Inf. Sci.* **2020**, *526*, 151–165.

18. Chunaev, P. Community detection in node-attributed social networks: A survey. *Comput. Sci. Rev.* **2020**, *37*, 100286.
19. Baroni, A.; Conte, A.; Patrignani, M. Efficiently clustering very large attributed graphs. In Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, Sydney, Australia, 31 July–3 August 2017; Volume 2017, pp. 369–376.
20. Zhou, Y.; Cheng, H.; Yu, J.X. Graph clustering based on structural/attribute similarities. *Proc. VLDB Endow.* **2009**, *2*, 718–729.
21. Cheng, H.; Zhou, Y.; Huang, X.; Yu, J.X. Clustering large attributed information networks: An efficient incremental computing approach. *Data Min. Knowl. Discov.* **2012**, *25*, 450–477.
22. Wang, C.; Pan, S.; Long, G.; Zhu, X.; Jiang, J. MGAE: Marginalized Graph Autoencoder for Graph Clustering. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; Volume 2017, pp. 889–898.
23. Gao, H.; Huang, H. Deep Attributed Network Embedding. In Proceedings of the IJCAI-ECAI-18, Stockholm, Sweden, 13–19 July 2018; Volume 18, pp. 3364–3370.
24. Reihanian, A.; Feizi-Derakhshi, M.R.; Aghdasi, H.S. Community detection in social networks with node attributes based on multi-objective biogeography based optimization. *Eng. Appl. Artif. Intell.* **2017**, *62*, 51–67.
25. Li, Z.; Liu, J.; Wu, K. A multiobjective evolutionary algorithm based on structural and attribute similarities for community detection in attributed networks. *IEEE Trans. Cybern.* **2017**, *48*, 1963–1976.
26. Li, Z.; Pan, Z.; Hu, G.; Li, G.; Zhou, X. Detecting semantic communities in social networks. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2017**, *100*, 2507–2512.
27. Qin, M.; Jin, D.; Lei, K. Adaptive community detection incorporating topology and content in social networks. *Knowl.-Based Syst.* **2018**, *161*, 342–356.
28. Yang, J.; McAuley, J.; Leskovec, J. Community detection in networks with node attributes. In Proceedings of the 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, 7–12 December 2013; Volume 2013, pp. 1151–1156.
29. Bojchevski, A.; Günnemann, S. Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 2018, pp. 2738–2745.
30. Wen, X.; Chen, W.N.; Lin, Y.; Gu, T.; Zhang, H. A maximal clique based multiobjective evolutionary algorithm for overlapping community detection. *IEEE Trans. Evol. Comput.* **2016**, *21*, 363–377.
31. Zhang, L.; Pan, H.; Su, Y. A mixed representation-based multiobjective evolutionary algorithm for overlapping community detection. *IEEE Trans. Cybern.* **2017**, *47*, 2703–2716.
32. Bello-Orgaz, G.; Salcedo-Sanz, S.; Camacho, D. A multi-objective genetic algorithm for overlapping community detection based on edge encoding. *Inf. Sci.* **2018**, *462*, 290–314.
33. Bai, X.; Yang, P.; Shi, X. An overlapping community detection algorithm based on density peaks. *Neurocomputing* **2017**, *226*, 7–15.
34. Ma, T.; Wang, Y.; Tang, M.; Cao, J.; Tian, Y. LED: A fast overlapping communities detection algorithm based on structural clustering. *Neurocomputing* **2016**, *207*, 488–500.
35. Whang, J.J.; Gleich, D.F.; Dhillon, I.S. Overlapping community detection using neighborhood-inflated seed expansion. *IEEE Trans. Knowl. Data Eng.* **2016**, *28*, 1272–1284.
36. Wang, X.; Liu, G.; Li, J. Overlapping community detection based on structural centrality in complex networks. *IEEE Access* **2017**, *5*, 25258–25269.
37. Deng, X.; Li, G.; Dong, M.; Ota, K. Finding overlapping communities based on Markov chain and link clustering. *Peer-to-Peer Netw. Appl.* **2017**, *10*, 411–420.
38. Li, Y.; Sha, C.; Huang, X.; Zhang, Y. Community detection in attributed graphs: An embedding approach. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 2018, pp. 338–345.
39. Al-Akhras, M.T. An Efficient Fuzzy K-Medoids Method. *World Appl. Sci. J.* **2010**, *10*, 574–583.
40. McAuley, J.J.; Leskovec, J. Learning to discover social circles in ego networks. *NIPS* **2012**, *2012*, 548–556.
41. Wei, H.; Pan, Z.; Hu, G.; Yang, H.; Li, X.; Zhou, X. Attributed network representation learning via DeepWalk. *Intell. Data Anal.* **2019**, *23*, 877–893.
42. Lancichinetti, A.; Fortunato, S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E* **2009**, *80*, 016118.
43. Huang, B.; Wang, C.; Wang, B. NMLPA: Uncovering Overlapping Communities in Attributed Networks via a Multi-Label Propagation Approach. *Sensors* **2019**, *19*, 260.
44. McDaid, A.F.; Greene, D.; Hurley, N. Normalized mutual information to evaluate overlapping community finding algorithms. *arXiv* **2011**, arXiv:1110.2515.
45. MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*; University of California Press: Berkeley, CA, USA, 1967; pp. 281–297.