Hindawi
Complexity
Volume 2021, Article ID 1376609, 10 pages
https://doi.org/10.1155/2021/1376609

WILEY | Hindawi

Research Article

A New Algorithm and Its Application in Detecting Community of the Bipartite Complex Network

Zhongyi Lei and Haiying Wang

School of Science, China University of Geosciences (Beijing), Beijing 100083, China

Correspondence should be addressed to Haiying Wang; whycht@126.com

Received 7 June 2021; Accepted 14 August 2021; Published 30 August 2021

Academic Editor: Sheng Du

The community division of bipartite networks is one frontier problem on the research of complex networks today. In this study, we propose a model of community detection of the bipartite network, which is based on the generalized suffix tree algorithm. First, extract the adjacent node sequences from the matrix of relation and use the obtained adjacent node sequences to build a generalized suffix tree; second, traverse the established generalized suffix tree to obtain the bipartite cliques; third, adjust the bipartite cliques; finally, dispose the isolated edges, get the communities, and complete the division of the bipartite network. This algorithm is different from the traditional community mining one since it uses edges as the community division medium and does not need to specify the number of the division of communities before the experiment. Furthermore, we can find overlapping communities by this new algorithm which can decrease the time complexity.

1. Introduction

In the natural and social fields, many complex systems can be expressed as networks composed of nodes connected by edges, such as the Internet, metabolic network, food chain network, neural network, communication and distributed network, logistics and supply chain network, industrial cluster network, and social organization network [1]. Among them, the bipartite network is one of the simplest complex networks. The bipartite network is composed of two types of node sets and they are connected, but there are no edges between any nodes of the same type.

With the development, people have discovered that many real networks have an important feature called community structure, that is, the entire network is composed of several communities, and these communities have structural features that are tight inside and loose outside [2]. Extending to the bipartite network, it can be considered that the heterogeneous nodes of different types within the same community are closely connected, while the connections of heterogeneous nodes between the communities are relatively sparse [3].

It is very important to study the theory of the bipartite network community mining. At the same time, we should have a deeper understanding of the complex network structure and the hidden laws and behavior characteristics of the network through the study of the community structure and then reasonably explain the network operation mode and make network predictions. This is the significance of this research. For example, in the social relationship network, real social groups can be displayed based on interests, occupations, regions, and backgrounds, so that character analysis, career recommendation, circle recommendation, friend recommendation, alumni discovery, and accurate advertising can be performed. In the citation network community, you can search and discover articles based on subject terms, authors, content, and units. We can make relevant recommendations based on user search terms or analyze the number of citations and quality to determine the impact factor or the design of the duplicate check algorithm, all of which need the support of network community theory. More and more practical problems need to rely on the community mining of the bipartite network for practice.

This is also the contribution we hope to make to real life by proposing the algorithm in this article.

In this study, we propose a community mining model of the bipartite network based on the generalized suffix tree algorithm. First, we use the node sequences extracted from the matrix of relation to build a generalized suffix tree; second, we traverse the generalized suffix tree to obtain the bipartite cliques; third, we form the initial communities according to the closeness after adjusting the bipartite cliques; finally, we dispose the isolated edges to obtain the result of final division. This model uses edges as a new division medium. Compared with the traditional community mining model with point division, this model can not only find more overlapping communities but also complete the community division of the bipartite network more efficiently.

2. Detecting Community from the Bipartite Network Based on Generalized Suffix Tree

2.1. Related Definitions. Use $G(U^X, U^Y, E)$ to represent a bipartite network, where U^X and U^Y are two different types of node sets in the network. There are m nodes in U^X, n nodes in U^Y, and E is the edge set. The adjacency matrix of the bipartite network is

$$\overline{M} = \begin{bmatrix} 0_{m*m} & M_{m*n} \\ \left(M^T\right)_{n*m} & 0_{n*n} \end{bmatrix}. \tag{1}$$

Since there is no connection between nodes of the same type, 0_{m*m} and 0_{n*n} are both zero matrices, and there are connections between nodes of different types, M_{m*n} and $(M^T)_{n*m}$ are both nonzero matrices. Since $\overline{M}$ is an adjacency matrix, M_{m*n} can be used to represent the basic characteristics of the entire bipartite network. We call M_{m*n} the matrix of relation of the bipartite network G.

The overall idea of the community division of the bipartite network is generally to divide the bipartite network $G(U^X, U^Y, E)$ through an algorithm to obtain c subgraphs $G_s(U_s^X, U_s^Y, E_s)$, $(s = 1, \ldots, c)$, in which there are $U_s^X \in U^X$, $U_s^Y \in U^Y$, $\cup_{s=1}^{c} U_s^X = U^X$, and $\cup_{s=1}^{c} U_s^Y = U^Y$. A good community division result shows that there are fewer connections between the c subgraphs and more connections within the same subgraph.

Modularity is an index that measures the quality of network community division in a numerical form. The greater the modularity value of a network community after division, the stronger the community relevance of the network, and the better the quality of community division. In this paper, we will take Murata's modularity [4] as the evaluation standard to measure the quality of community division.

Let E be the edge set of the bipartite network, and its two different types of node sets U^X and U^Y. Let U_l and U_m be different types of node sets within a community with $U_l \in U^X$ and $U_m \in U^Y$. Denote $A(i, j)$ be an element in the i^{th} row and j^{th} column of the matrix of relation of the bipartite network. And we let $A(i, j) = 1$ when there is a connection between the i^{th} node in U^X and the j^{th} node in U^Y; otherwise, it is 0.

First, we define the proportion of the connection between the node sets U_l and U_m in all the connections of the bipartite network. Let e_{lm} be an element in the l^{th} row and m^{th} column as follows:

$$e_{lm} = \frac{1}{2|E|} \sum_{i \in U_l} \sum_{j \in U_m} A(i, j). \tag{2}$$

Then, the sum of the elements in the l^{th} row α_l can be obtained as

$$\alpha_l = \sum_m e_{lm} = \sum_{i \in U_l} \sum_{j \in U_m} A(i, j). \tag{3}$$

Based on the definitions above, the Murata modularity is defined as Q_B by

$$Q_B = \sum_l Q_{Bl} = \sum_l (e_{lm} - \alpha_l \alpha_m), \tag{4}$$

where Q_{Bl} represents the strength of the relationship between community m and community l. Obviously, the larger the value of Q_B, the stronger the community relevance of the detected bipartite network and the better the community division effects.

2.2. Algorithm Design. A suffix tree is a rooted directed tree and represented any string S of length m with exactly m leaves numbered 1–m. Any internal node other than the root in the suffix tree has a minimum of two children, and each edge is labeled with a nonempty substring of S.

Any labels of any two edges out of a node must begin with different characters. For any leaf i in the suffix tree, the concatenation of the edge-labels on the path from the root to leaf i exactly spells out the suffix of S that starts at position i. A generalized suffix tree is a suffix tree that combines the suffixes of a set $\{S_1, \ldots, S_n\}$ of strings [5]. The following describes the specific six steps of a new algorithm in this study.

Step 1. Convert the bipartite network into a matrix of relation and then extract the adjacent node sequences.

Suppose that $G(U^X, U^Y, E)$ is the bipartite network shown in Figure 1(a), and it has two types of nodes U^X and U^Y, among which U^X has four nodes $U^X = \{x_1, x_2, x_3, x_4\}$ and U^Y has five nodes $U^Y = \{y_1, y_2, y_3, y_4, y_5\}$. In this way, we can use a $4 * 5$ matrix of relation to express this bipartite network. When there is a connection between the i^{th} node in U^X and the j^{th} node in U^Y, the value of the element in the i^{th} row and j^{th} column is 1; otherwise, it is 0, as shown in Figure 1(b).

Definition 1 (Adjacent node sequence). After the bipartite network $G(U^X, U^Y, E)$ is transformed into a matrix of relation, the adjacent node sequence $S_i = (j^1, j^2, \ldots, j^k)$ of the node i can be extracted from the matrix with $i \in U^X$.

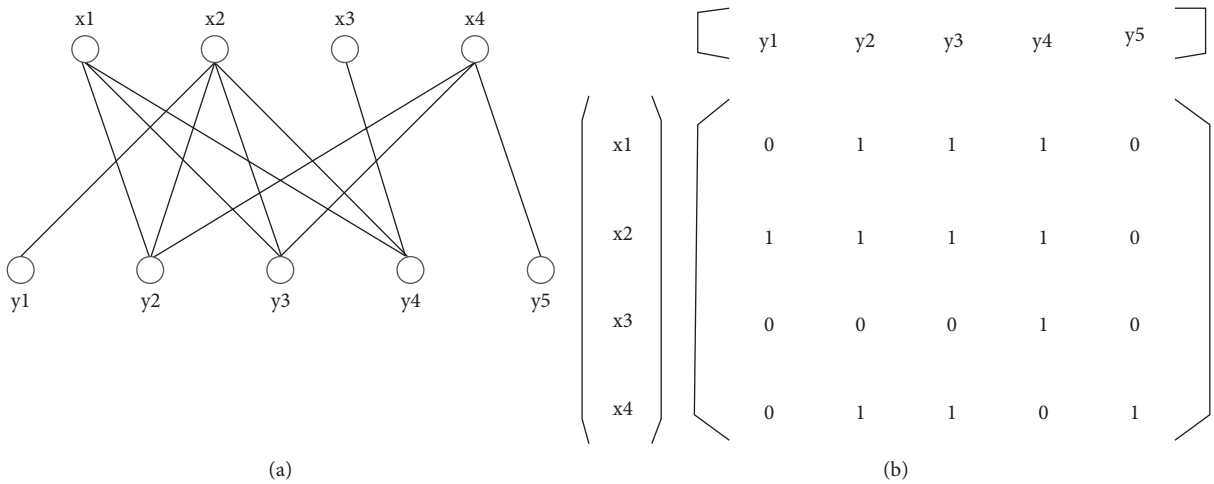


Figure 1: Bipartite network and its relation matrix.

From the matrix of relation, we can get the adjacent node sequences of U^X as follows:

$$\begin{aligned} S_1 &= (2,3,4),\\ S_2 &= (1,2,3,4),\\ S_3 &= (4),\\ S_4 &= (2,3,5). \end{aligned} \tag{5}$$

Step 2. Create a generalized suffix tree based on the adjacent node sequences.

The generalized suffix tree is constructed from the adjacent node sequences obtained in Step 1. Mark each branch with a nonempty substring to reflect the elements in each adjacent node sequence and in each suffix subsequence of them. A branch of the generalized suffix tree represents an adjacent node sequence, and the same prefixes of sequences share the same branch. Then, reserve a set $B = \{p|$the adjacent node sequences S_p through a node in the generalized suffix tree.

According to the above, we can get the generalized suffix tree of Figure 1(a) as shown in Figure 2.

Step 3. Extract the initial bipartite cliques.

Definition 2 (Bipartite clique). Using the nonempty substring on the branch of the generalized suffix tree and the set B stored by the node, each node can be represented as a bipartite clique $C_i = (U_i^X, U_i^Y)$ with $U_i^X \in U^X$ and $U_i^Y \in U^Y$, in which U_i^X is the element in set B stored by each node i, and U_i^Y is a set of nonempty substrings on the path from the root node of the generalized suffix tree to node i.

The initial bipartite cliques can be extracted from the constructed generalized suffix tree. From Figure 2, the initial bipartite cliques can be obtained as follows:

$C_1 = (\{2\},\{1,2,3,4\})$, $C_2 = (\{1,2,4\},\{2,3\})$, $C_3 = (\{1,2\},\{2,3,4\})$, $C_4 = (\{4\},\{2,3,5\})$, $C_5 = (\{1,2,4\},\{3\})$, $C_6 = (\{1,2\},\{3,4\})$, $C_7 = (\{4\},\{3,5\})$, $C_8 = (\{1,2,3\},\{4\})$, and $C_9 = (\{4\},\{5\})$.

Step 4. Adjust the initial bipartite cliques.

Definition 3 (Adjustable bipartite cliques). Let (X_1, Y_1) and (X_2, Y_2) be both initial bipartite cliques. If $(X_1 \cup X_2, Y_1 \cap Y_2) = (Z, Y_2)$, where $Z = X_1 \cup X_2$, it is said that (X_1, Y_1) and (X_2, Y_2) are adjustable bipartite cliques. The specific adjustment operation is to adjust (X_2, Y_2) to (Z, Y_2) with $Z = X_1 \cup X_2$ and to keep (X_1, Y_1) no change.

In simple terms, the aim of this definition is to convert two adjustable bipartite cliques (X_1, Y_1) and (X_2, Y_2) into bipartite cliques (X_1, Y_1) and $(X_1 \cup X_2, Y_2)$ after adjustment.

For example above, $C_1 = (\{2\},\{1,2,3,4\})$ and $C_2 = (\{1,4\},\{2,3\})$ can be adjusted to $C_1 = (\{2\},\{1,2,3,4\})$ and $C_2 = (\{1,2,4\},\{2,3\})$ according to Definition 3.

After adjusting the eight initial bipartite cliques obtained in Step 3, the following three bipartite cliques can be obtained $C_1 = (\{1,2\},\{2,3,4\})$, $C_2 = (\{2\},\{1,2,3,4\})$, and $C_3 = (\{4\},\{2,3,5\})$. The bipartite networks corresponding to these three bipartite cliques are shown in Figure 3.

Step 5. Combine the bipartite cliques to form the initial communities based on the tightness.

According to Figure 3, the adjusted bipartite clique only requires edge connections between heterogeneous nodes, while the community requires close connections between heterogeneous nodes within the community. Therefore, to get the initial communities, the adjusted bipartite cliques need to be further processed. Here, we choose the method of merging according to the tightness to turn the bipartite cliques into the initial communities.

Definition 4 (Tightness). Set bipartite cliques $C_1 = (U_1^X, U_1^Y)$, $C_2 = (U_2^X, U_2^Y)$, and $C = C_1 \cap C_2$, where $|C|$ represents the number of inner edges of the bipartite clique C. The tightness written by D of C_1 and C_2 is defined as

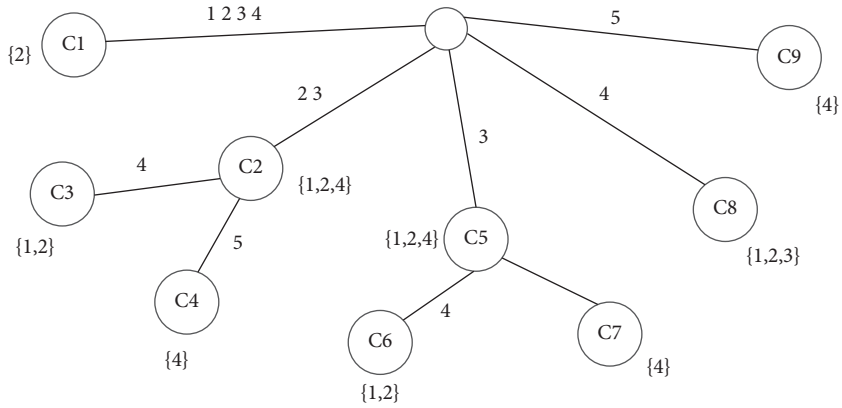


Figure 2: The generalized suffix tree.

$$D = \frac{2 \times |C| - \left[\left|C_1 - C\right| + \left|C_2 - C\right|\right]}{\min\left(\left|C_1\right|, \left|C_2\right|\right)}. \quad (6)$$

In the process of merging bipartite cliques according to the tightness, the median value of the tightness calculated for the first time is used as the threshold ε of this experiment. If the closeness between the two dichotomy groups is greater than the threshold ε, they are merged. If the maximum compactness of the existing two bipartite cliques is less than the threshold ε, the merging is stopped. We call the resulting bipartite cliques the initial communities.

For example, above in Figure 1(a), its initial communities of the bipartite network obtained are $\{(1,2), (1,3), (1,4), (2,3), (2,2), (2,4)\}$, $\{(2,1)\}$, $\{(3,4)\}$, and $\{(4,5), (4,2), (4,3)\}$. Note that the edges are used as the mediums to divide the communities, and the results are that the edges in a certain set which belongs to the same community, rather than some points forming a community. To reflect the results of the division on the bipartite graph, different communities are marked with different colors in Figure 4.

Step 6. Division of isolated edges.

After the division in Step 5, there may still be some edges that are not classified into a certain community because the tightness does not reach the threshold. We call these edges the isolated edges. The dataset above does not involve this problem. But it is necessary to divide the isolated edges by calculating the tightness of an edge to the communities if this situation exists during dividing other datasets.

Definition 5 (Tightness of isolated edges). Let e be an isolated edge that is not divided into any communities. First, transform the isolated edge e into a bipartite clique $\{e\}$ with only one edge; second, calculate the tightness between the bipartite clique $\{e\}$ and each existing initial communities according to formula (6), and then, the isolated edge e is divided into the community which has the highest tightness with $\{e\}$. This completes the division of communities with the isolated edges.

Through the six steps above, we can complete the community detection of the bipartite network and the initial communities can be obtained. The pseudocode of this new algorithm is as follows (Algorithm 1).

3. Data Experiments

In order to verify the performance and effect of the algorithm of community detection of the bipartite network based on the generalized suffix tree, the algorithm is now used for data experiments on the computer. The experimental environment is 8 GB memory, Intel Core i7 CPU, Windows 10 operating system, implemented using python programming.

3.1. Function Test

3.1.1. Network Test with Obvious Community Structure. In order to test that the generalized suffix tree algorithm has a logical community division effect, we now design a bipartite network with an obvious community structure, as shown in Figure 5. Obviously, this bipartite network can be divided into two communities $\{x_1 - x_3, y_1 - y_3\}$ and $\{x_4 - x_6, y_4 - y_6\}$. After extracting the matrix of relation and adjacent node sequences of the network, enter the program and run the program to divide the communities.

The result is shown in Figure 6. Different communities are distinguished by lines of different colors. Here, a certain fixed node is divided into the community with the highest degree of connection with it, in order to obtain a clearer node division effect.

The results show that the community mining algorithm based on the generalized suffix tree divides the bipartite network into two expected initial communities.

From this, we can draw the conclusion that the generalized suffix tree algorithm can draw a community division result that is consistent with the objective and actual conclusion when dividing a bipartite network with an obvious community structure.

Designing several similar experiments, use this algorithm to divide a bipartite network with a more obvious community structure. These experiments can get satisfactory results and consistent with expectations.

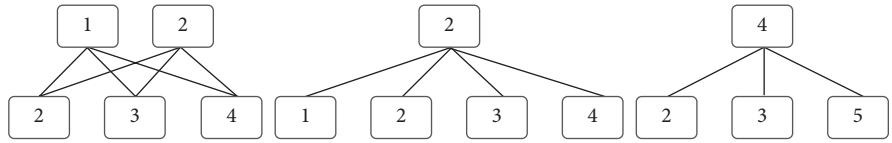


Figure 3: Adjusted bipartite cliques.

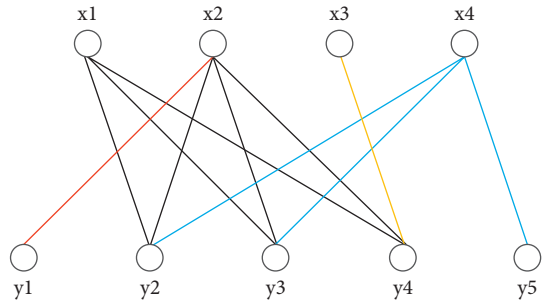


Figure 4: The results of the division on the bipartite graph.

Through the above experiments, we can conclude that the algorithm of community detection based on the generalized suffix tree has a logically reasonable community division effect, can well divide the bipartite network with obvious community structure, and obtain the expected division result. This result reflects the reliability of the algorithm for dividing the bipartite networks with obvious community structures and can be extended to the bipartite networks with less obvious community structures.

3.1.2. Overlapping Network Test. In most traditional community detection models, the community division uses the nodes as a medium to operate. However, in order to better discover overlapping communities and avoid the ambiguity of the connection relationship in the community, this algorithm uses some edges as the main body of the medium and experimental operation to divide the community. To test that this model can better discover overlapping communities, this experiment is specially designed. There is a bipartite network as shown in Figure 7. The network has an obvious community structure. First, the bipartite network can be divided into two communities $\{x_1 - x_2, y_1 - y_3\}$ and $\{x_3 - x_4, y_3 - y_5\}$. Second, y_3 is an overlapping node which belongs to two different communities.

Now, we extract the adjacent node sequences of the bipartite network and enter the program above to divide the community, and the result is shown in Figure 8. This algorithm divides the bipartite network into two expected communities, and the overlapping communities can be obtained, that is, y_3 is an overlapping node which belongs to two different communities.

From the above experiment, we can conclude that this algorithm uses some edges as the main mediums of community division, which can successfully complete the division of communities based on ensuring a clear and stable community structure, and can well find overlapping communities in the bipartite network.

3.2. Dataset Experiment of Davis Southern Club Women. Of course, whether the algorithm model has a good division effect cannot be completely defined by logical judgment and analogy promotion. The real division quality should be judged and compared based on the value of the modularity.

This data experiment uses the Southern Club Women dataset collected by Davis in the 1930s. The dataset describes the correspondence between 18 women and 14 club events in the Southern Women's Club of Mississippi. It can be represented by a bipartite network diagram, in which one type of node represents women and the other type of nodes represents events. If a woman participates in an event, there will be an edge connecting the node represented by this woman and the node represented by this event. There is no connection between the nodes represented by each woman and between the nodes represented by each event. As shown in Figure 9, different types of nodes are represented by different colors. $W_1 - W_{18}$ in the first row represent 18 woman nodes and $E_1 - E_{14}$ in the second row represent 14 event nodes.

Extract the adjacent node sequences of the bipartite network, enter the program, and use the generalized suffix tree algorithm to divide the community of the bipartite network. The result of the division is shown in Figure 10. The edges of different communities are marked with different colors.

Next, check the modularity. In order to discover as many overlapping communities as possible, as long as the same node is connected to edges from different communities, this node is defined as an overlapping node, and it belongs to different communities that are connected to it at the same time. Two communities can be obtained after community division using the generalized suffix tree algorithm: {woman 2, 12, event 1, 2, 6, 13, 14} and {woman 1 − 18, event 1 − 14}. This study will take Murata's bipartite modularity as the final evaluation standard to measure the quality of community division. The experimental result of modularity is 0.1563, indicating that this model is implementable.

Input: The matrix of relation of a bipartite graph $G(U^X, U^Y, E)$.
Output: The initial communities.

Begin
// **Step 1: Get the adjacent node sequences.**
for $i \in U^X$ **do**
  Calculate the adjacent node sequences $S_i = (j^1, j^2, \ldots, j^k)$.
  Let S be the set of all adjacent node sequences, that is, $S = (S_1, S_2, \ldots, S_n)$.
end for

// **Step 2: Integrate the adjacent node sequences into a linked list to facilitate the establishment of a generalized suffix tree.**
Create a node class with two attributes, *node .val* and *node .next*.
for $n \in S$ **do**
  Convert $S_i = (j^1, j^2, \ldots, j^k)$ to $S_i = j^1 \longrightarrow j^2 \longrightarrow \cdots \longrightarrow j^k$.
end for
Create a suffix linked list according to the node linked list:
for $n \in S$ **do**
  Convert each $S_i = j^1 \longrightarrow j^2 \longrightarrow \cdots \longrightarrow j^k$ to $S(i_m) = j_m^1 \longrightarrow j_m^2 \longrightarrow \cdots \longrightarrow j_m^k$, $(1 < m \leq k)$.
end for

// **Step 3: Establish a generalized suffix tree according to the linked list of adjacent node sequences.**
Create a tree_node class with two attributes, tree_node.val and tree_node.next. Since the number of child nodes of the generalized suffix tree is uncertain, tree_node.next is an array of tree_node, which stores all its child nodes.
Root node is null node.
Insert all the linked lists of adjacent node sequence into the generalized suffix tree, the process is:
for $i \in S$ **do**
  $p = j_i^1$
  while p **do**
    if $p \notin$ root.next **then**
      Insert the linked lists from p to the Root node.
    else
      Root = p ; $p = p$. next.
    end if
  end while
end for

// **Step 4: Get the bipartite cliques through the generalized suffix tree.**
for $i \in$ tree_node **do**
  if i is a leaf node, that is, tree_node.next =, or i is a branch node, that is, |tree_node.next| > 1 **then**
    Create the bipartite clique $C = \langle U^X, U^Y \rangle$.
  end if
end for

// **Step 5: Adjust the bipartite cliques.**

// **Step 6: Merging the bipartite cliques to form the initial communities.**
while true **do**
  Calculate the tightnesses of bipartite cliques: $R = R(C_i, C_j)$, $(i \neq j)$.
  if $\max(R) >$ threshold ε **then**
    Merge the bipartite cliques.
  end if
end while

// **Step 7: Adjust the isolated edges and divide them into communities.**
Taking the isolated edge as bipartite cliques, and calculate the tightness R.
Divide the isolated edges into the communities with the highest tightness.
End

Algorithm 1: Detecting community from the bipartite network based on generalized suffix tree.

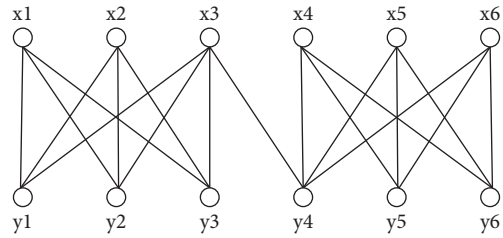


Figure 5: A bipartite network with an obvious community structure.

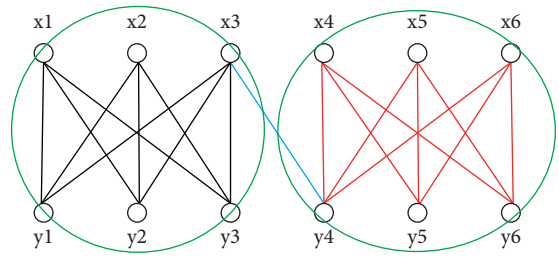


Figure 6: The results of the division on the bipartite network with an obvious community structure.

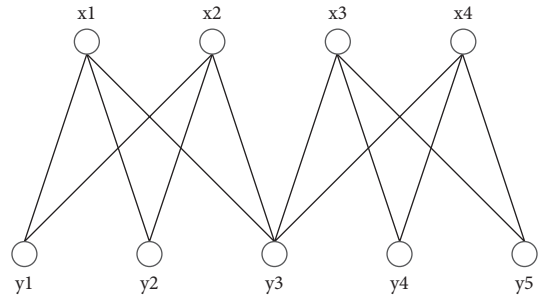


Figure 7: A bipartite network with overlapping communities.

3.3. Disease-Gene Network Experiment. In this subsection, we will next introduce a disease-gene network experiment. Cell is the basic unit of human body, composed of cell membrane, cytoplasm, and cell nucleus. All the genetic information of the cell is recorded on the DNA gene fragments of the nucleus. Genetic diseases are caused by mutations in genes of multiple DNA fragments. We call genes related to genetic diseases as disease genes [6].

A document of "The human disease network" in PNAS abstracts the relationship between existing genetic diseases and disease genes into a network structure [7] and provides a graph theory-based research platform for studying various disease characterizations and the relationship between genes [8–10]. As shown in Figure 11(a), genetic diseases and gene nodes form a common complex network structure. If a disease gene mutation can directly or indirectly cause a certain disease, then use an edge to connect them. Since there is no connection relationship between nodes of the same type, ordinary complex networks can be transformed into bipartite network representation, as shown in Figure 11(b). The 19 nodes on the left represent all known genetic diseases, and the right 19 nodes represent all known disease genes, and the size of the node is proportional to the degree of the node. The data on genetic diseases, disease genes, and the relationship between them are all from the OMIM database and document [7]. According to the information provided in Figure 11, we can study the interaction and internal connections between these 19 genetic diseases and 19 disease genes. Based on this theory, it can be extended to study the relationship between all known disease genes and hereditary diseases and draw conclusions of universal significance.

Now, we begin to perform the data experiment. First, use the algorithm of community detection based on the generalized suffix tree to divide the abovementioned disease-gene bipartite network, and we obtain the results shown in Figure 12. From Figure 12, we can see that the algorithm divides the edges in the disease-gene bipartite network into five types and different types of edges are marked with different colors. In order to discover as many overlapping communities as possible, this node is defined as an overlapping node as long as the same node is connected to edges from different communities at the same time.

By this way above, five communities can be obtained:
$\{\text{diseases } 1-5, \text{genes } 1-3\}$,
$\{\text{diseases } 6, 7, 9, \text{genes } 1-5, 8, 10-16\}$,

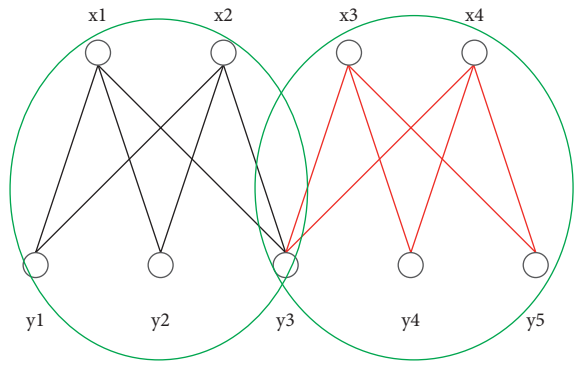


Figure 8: The results of the division on the bipartite network with overlapping communities.

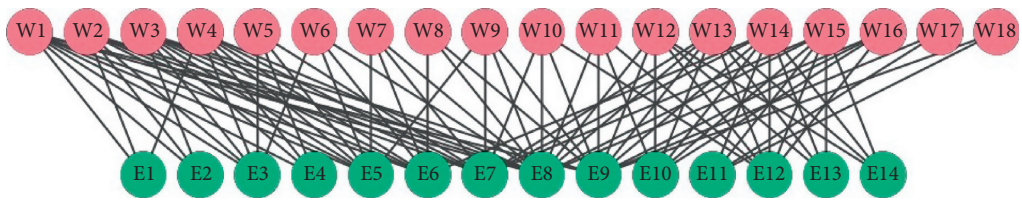


Figure 9: Bipartite network of Southern Club Women.

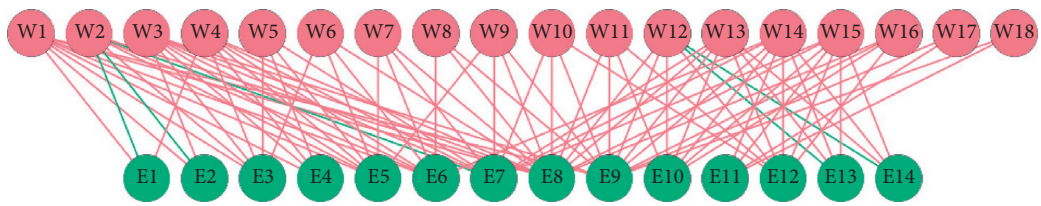
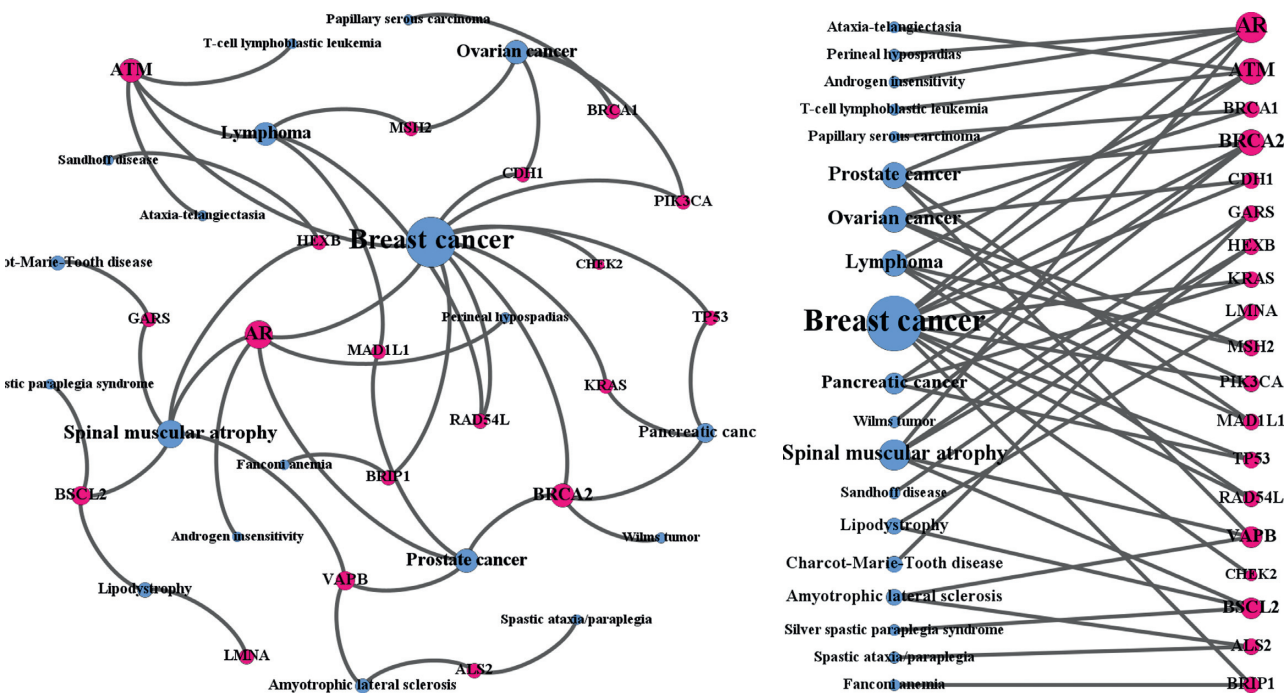


Figure 10: The results of the division on the bipartite network of Southern Club Women.



(a) (b)

Figure 11: Disease-gene network.

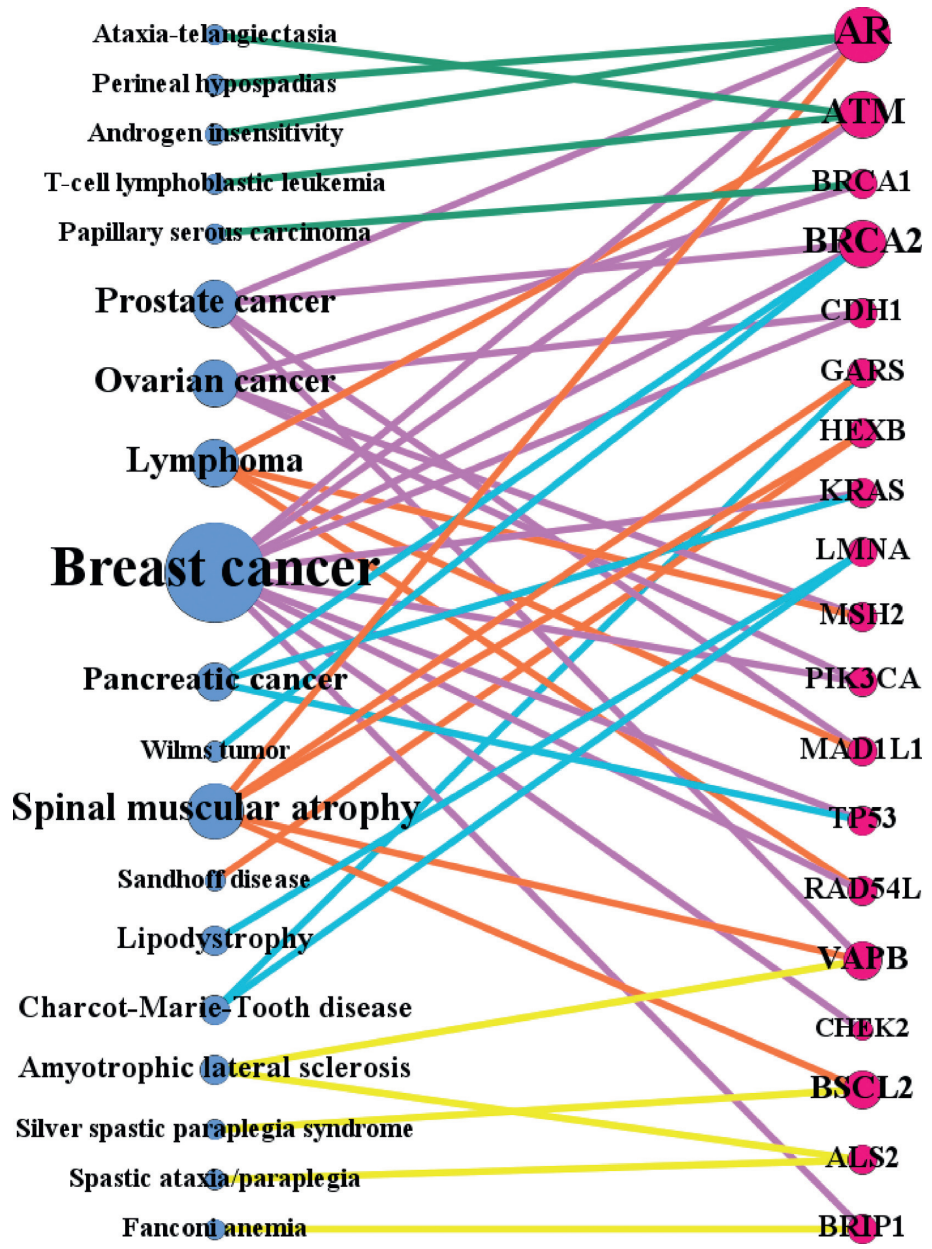


Figure 12: The results of the division on the disease-gene network.

{diseases 8, 12, genes 1, 2, 6, 7, 10, 12, 14, 15, 17},
{diseases 10, 11, 14, 15, genes 4, 6, 8, 9, 13}, and
{disease 16 − 19, genes 15, 17 − 19}.

Using Murata' bipartite modularity as the evaluation standard to measure the quality of community division, the modularity experiment result is 0.0659, indicating that this model is scientific and feasible, and the division results of the model can be further applied to other diseases and gene research.

4. Conclusion

In this study, we propose a new method of detecting community from the bipartite complex network based on generalized suffix tree. The ordinary bipartite network is represented by a generalized suffix tree structure, and then, the community division of the general bipartite network is completed by traversing the suffix tree and performing subsequent operations. Above all, this model does not need to specify the number of communities that need to be divided in advance, and it can complete the community mining of the bipartite network with lower time complexity. This new community mining model is different from the traditional ones as it puts forward the viewpoint of community division using the edge in the network as a medium, implements it, and obtains a satisfactory division result. Other applications of these results are worthy of further exploration and expansion in follow-up research.

Data Availability

The data for "Davis Southern Club Women" are available in the UCINET IV datasets: http://vlado.fmf.uni-lj.si/pub/networks/data/Ucinet/UciData.htm#davis and the "Disease-Gene" dataset analyzed is available in the published article: The human disease network (http://www.pnas.org/cgi/doi/10.1073/pnas.0701361104). The list of disorders, disease genes, and associations between them is publicly available on the OMIM database (https://omim.org/).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (11701530) and Fundamental Research Funds for the Central Universities (2020003).

References

[1] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 23, pp. 8577–8582, 2006.

[2] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002.

[3] P. Zhang, J. Wang, X. Li, M. Li, Z. Di, and Y. Fan, "Clustering coefficient and community structure of bipartite networks," *Physica A: Statistical Mechanics and Its Applications*, vol. 387, no. 27, pp. 6869–6875, 2008.

[4] T. Murata, "Modularity for bipartite networks," *Data Mining for Social Network Data, Annals of Information Systems*, Springer, vol. 12, no. 7, , pp. 109–123, Berlin, Germany, 2010.

[5] Z. Liang, K. Hu, Y. Ning, and D. Yisheng, "An efficient index structure for XML based on generalized suffix tree," *Information Systems*, vol. 32, no. 2, pp. 283–294, 2007.

[6] W. Chen, J. Lu, and J. Liang, "Research in disease-gene network based on bipartite network projection," *Complex Systems and Complexity Science*, vol. 6, no. 1, pp. 13–19, 2009.

[7] K.-I. Goh, M. E. Cusick, D. Valle, B. Childs, M. Vidal, and A.-L. Barabasi, "The human disease network," *Proceedings of the National Academy of Sciences*, vol. 104, no. 21, pp. 8685–8690, 2007.

[8] H. Ada, A. F. Scott, A. Joanna, C. A. Bocchini, and V. A. McKusick, "Online mendelian inheritance in man (OMIM), a knowledgebase of human genes and genetic disorders," *Nucleic Acids Research*, vol. 33, pp. 514–517, 2005.

[9] M. A. Pujana, J.-D. J. Han, L. M. Starita et al., "Network modeling links breast cancer susceptibility and centrosome dysfunction," *Nature Genetics*, vol. 39, no. 11, pp. 1338–1349, 2007.

[10] P. A. Futreal, L. Coin, M. Marshall et al., "A census of human cancer genes," *Nature Reviews Cancer*, vol. 4, no. 3, pp. 177–183, 2004.