A New Approach to Detect Communities in Multi-Weighted Co-authorship Networks

Evelyn Perez Cervantes  
School of Systems Engineering  
National University of St Augustin, Peru  
epcervantes7@gmail.com

Jesús P. Mena-Chalco  
Department of Computer Science  
University of São Paulo, Brazil  
jmena@vision.ime.usp.br

Abstract— Co-authorship graphs related to publications (bibliographical production) are generally represented by undirected graphs, where each edge maintains a single value for publications produced by the authors. A variation of this type of graph is related to use of different weights associated with different types of publications. In this context, within the field of scientometrics, ‘journal papers’ or ‘books’ may have a higher priority than the ‘conference papers’, and the ‘conference papers’ might have higher priority than the ‘extended abstract’. In this paper, we present a simple weight combination to detect communities in co-authorship networks, which allows simultaneous consideration of multiple types of collaborations. Our preliminary results show a good performance in the detection of communities considering real bibliographical production graphs.

Keywords-Co-authorship network; community structure detection; multi-weighted graph

I. INTRODUCTION

Recently, the algorithms for community structure detection have evolved and each time they use a greater amount of information. The first and most simple case is the communities detection in a binary and undirected co-authorship network. If two authors co-write a scientific paper, a unitary edge is created. Thus, the resulting graph is represented as an undirected unit-weighted graph. The second category is made up of binary and directed co-authorship networks, every edge in the undirected network is replaced by two, symmetrical and directed edge, the resulting graph is represented as a directed unit-weighted graph. This type of graph is commonly used to measure author prestige, but the binary graph does not represent co-authorships according to reality, so the most recent studies use weighted directed co-authorship networks, the weight of an edge is given by the number of co-authorship between the scientists who connect [1], [2], some of these studies in social networks do not use all information available in databases which leads to an unrealistic representation and detection of communities.

In this work Newman’s algorithm [3] is used to detect community structure within real weighted networks, in order to make a more realistic representation and detection of communities was added the multi-weighted concept on the edges, which means that instead of representing the weight of an edge (the relationship between co-authors/scientists) with a single value, it is represented by multiple values which have different weightings depending on the type of bibliographical production.

This algorithm ensures keep up an special feature of social networks called community structure, the natural division of network nodes into groups within which the network connections are dense, with only a smaller number of edges between vertices of different groups as Figure 1.

II. METHODOLOGY

There are several proposals [4] [5] to detect communities in graphs but we have not found any proposals that work with multiple weights in their edges or in this case with multiple types of bibliographical publication.

For example, if there are two scientists who co-wrote 10 papers where 5 of them were published as ‘chapter books’, other 3 of them were published in ‘conference proceedings’ and the other 2 were published in another type of publication.

It is natural to assign different priority levels to each group, e.g. the first group may have a weight of 3, and the next group a weight of 2 and the last group may have a weight of 1. The priority level is assigned arbitrarily depending on the relevance/importance of publications in

Figure 1: Network with community structure. In this case there are three communities locked inside the large dashed circles, which have dense internal links but between which there are only a lower density of external links.
scientometrics\textsuperscript{1}. It is possible to calculate the combination value for the weight of the edge, using a single operation as $5 \times 3 + 3 \times 2 + 2 \times 1$. Therefore, instead of using 10 as the weight of the edge, it is used 23. This simple example shows our idea to detect communities in multi-weighted networks.

This way of calculating the weights for edges allows us to make a more realistic representation of co-authorship, because not all bibliographical publications have the same importance or relevance in scientometrics. The real weight (rw) of an edge is calculated by:

$$rw(i,j) = M_1(i,j) \times p_1 + \ldots + M_k(i,j) \times p_k$$

(1)

where $p_k$ is the priority level measure for bibliographical productions of type $k$, and $M_k(i,j)$ represents the number of bibliographical productions between the author $i$ and the author $j$ for publications of type $k$.

In order to detect communities, after been calculated the real weights for all edges, we have used a modification of the Newman’s Shortest-path betweenness algorithm \textsuperscript{3}. This procedure allows determining the contribution of each edge used in the betweenness score.

Before describing the algorithm is useful to recall what betweenness mean, in graphs theory betweenness is a centrality measure of a edge (there is also vertex betweenness, which is not discussed here because it is not used in this manuscript) within a graph. Edges that occur on many shortest paths between other vertices have higher betweenness than those that do not (see Figure 2).

![Figure 2](image1.png)

Figure 2: Finding the shortest paths between all pairs of vertices and counts how many of the shortest paths pass along each edge, the edge in red is the one that has more occurrences or the highest betweenness score.

The algorithm to detect communities in a network is summarized as follows:

1) Calculate betweenness scores for all edges in the network;
2) Find the edge (or edges) with the highest score and remove them from the network;
3) Recalculate betweenness score for all remaining edges;
4) Repeat from step 2.

\textsuperscript{1}Scientometrics is the science of measuring and analysing science. In practice, scientometrics is often done using bibliometrics which is a measurement of the impact of (scientific) publications.

A. Shortest-Path

Shortest-path of vertex and edges, called ‘geodesic’ too, links two given vertices. The shortest path may not be unique for a pair or vertex as in Figure 3. There are several Shortest-path between vertex A and vertex B. In order to calculate the shortest-path given two vertex $i$ and $j$ have been used the following algorithm extracted from \textsuperscript{2}:

1) Assign vertex $j$ distance zero ($d = 0$), indicating the target vertex.
2) For each vertex $k$ whose distance is $d$, follow each edge to the neighbor vertex $l$, if $l$ has not already been assigned a distance, assign it distance $d + 1$. Declare $k$ to be a predecessor of $l$.
3) If $l$ has already been assigned distance $d+1$, then there is no need to do this again, but $k$ is still declared a predecessor of $l$.
4) Set $d = d + 1$.
5) Repeat from step 2 until there are no unassigned vertices left.

The shortest path (if there is one) from $i$ to $j$ is the path obtained by stepping from $i$ to its predecessor, and then to the predecessor of each successive vertex until $j$ is reached. If a vertex has two or more predecessors, then there are two or more shortest paths, each of which must be followed separately if we wish to know all shortest paths from $i$ to $j$. To help in the calculations, a queue (a first-in/first-out buffer) is used as in standard breadth-first search.

The output of the described algorithm is a set of shortest trees, one for each vertex. Figure 4(a) shows a shortest tree for the vertex $S$ with a unique shortest path for all other vertices, but in real networks in most cases has more than one shortest tree as Figure 4(b).

B. Shortest-Path Betweenness

Finding Shortest-Path Betweenness may be simple if all shortest tree were similar to Figure 4(a), with only a unique shortest path for all other vertices. In this case, we only need work upwards, assigning to each edge a score equal 1 plus the sum of the scores on the neighboring edges immediately below it, repeating the process for all vertices and finally

![Figure 3](image2.png)

Figure 3: the shortest path between vertex A and vertex G, the is not a unique shortest path.
summing the scores of all shortest trees. But this is only an ideal case because we need work with shortest trees with more than one shortest path for all other edges, first we must calculate the number of paths from the source to each other vertex (numbers on vertices), and then these are used to weight the path counts appropriately. Therefore, the algorithm need some steps more, the algorithm Shortest-Path Betweenness used our work is described in [6] and have de following steps:

1) The initial vertex \( s \) is given distance \( d_s = 0 \) and a weight \( w_s = 1 \);
2) Every vertex \( i \) adjacent to \( s \) is given distance \( d_i = d_s + 1 \), and weight \( w_i = w_s = 1 \);
3) For each vertex \( j \) adjacent to one of those vertices \( i \) we do one of three things:
   - If \( j \) has not yet been assigned a distance, it is assigned distance \( d_j = d_i + 1 \) and weight \( w_j = w_i \).
   - If \( j \) has already been assigned a distance and \( d_j = d_i + 1 \), then the vertex’s weight is increased by \( w_i \), that is \( w_j = w_j + w_i \).
   - If \( j \) has already been assigned a distance and \( d_j < d_i + 1 \), we do nothing.
4) Repeat from step 3 until no vertices remain that have assigned distances but whose neighbours do not have assigned distances.

This algorithm can be implemented most efficiently using a queue or first-in/first-out as in breadth-first search.

The weight on a vertex \( i \) represents the number of distinct paths from the source vertex to \( i \). These weights need to be calculated because they constitute edge betweenness, because if two vertices \( i \) and \( j \) are linked, with \( j \) farther than \( i \) from the source \( s \), then the fraction of a geodesic path from \( j \) through \( i \) to \( s \) is given by \( w_i/w_j \). Thus, to calculate the contribution to edge betweenness from all shortest paths starting at \( s \), was used the following steps:

1) Find every ‘leaf’ vertex \( t \), i.e., a vertex such that no paths from \( s \) to other vertices go through \( t \);
2) For each vertex \( i \) neighboring \( t \) assign a score to the edge from \( t \) to \( i \) of \( w_i/w_t \);
3) Starting with the edges that are farthest from the source vertex slower down in a diagram such as Figure 4 work up towards \( s \). To the edge from vertex \( i \) to vertex \( j \), with \( j \) being farther from \( s \) than \( i \), assign a score that is 1 plus the sum of the scores on the neighboring edges immediately below it (i.e., those with which it shares a common vertex), all multiplied by \( w_i/w_j \);
4) Repeat from step 3 until vertex \( s \) is reached.

Repeating this process for all vertices and summing the resulting scores on the edges gives us the total betweenness for all edges.

C. Weighted Networks

The real collaboration networks that we need to evaluate are weighted, the authors who have written many papers together will know one another better, on average, than those who have written few papers together. ‘co-authorship frequency’ [1] consist of the sum of all articles co-authored by author \( i \) and \( j \). This sum gives more weight to the edge that connects these authors, and the edges of author who co-publish more papers together have more weight. The number of publications between two authors is the weight in each edge, and the algorithms previously described have used unweighted networks. To solve this problem, Newman [7] introduces an algorithm that generalizes the shortest-path betweenness to works with weighted network which is described following:

Firstly a mapping from the weighted network to a multigraph (see Figure 5) must to be done. The edges are the same in the original weighted network and in the new multigraph. The network with information of authors is a weighted network with integer weights then each weight edge is replaced by \( n \) parallel edges of unit weight and the algorithm applied in normal unweighted network, now will be applied in the resulting multigraph but with a little modification the betweenness of each of the parallel edges is equal to the betweenness of the corresponding edge on the simple graph, divided by the number of parallel edges finally remove the edge with the highest resulting score, recalculate the betweennesses, and repeat.

![Multigraph](image-url)

Figure 5: Multigraph is a graph with more than only one edge that links a pair of vertex.
D. Modularity

The modified algorithm works with networks where the final numbers of communities are not known a priori. To determine how well the algorithm performs, a measure called modularity is used which measures the fraction of the edges in the network that connect vertices of the same community, i.e. within-community edges, minus the expected value of the same quantity in a network with the same community divisions but random connections between the vertices [3].

Considering a division in k communities, a symmetric matrix $M$ is defined of $k \times k$ whose elements $M_{ij}$ represent the fraction of edges of the network that link vertices of community $i$ with vertices of community $j$. The trace of the matrix $e$, $\text{Tr}(e) = \sum_{i} e_{ii}$, represent the edges of the network that links the vertices in the same community. Then if the division is good the value of trace is high.

A row (or column) sum is defined as $a_i = \sum_j e_{ij}$ which represents the fraction of edges that link to community $i$. Then, the modularity is define as:

$$Q = \sum_i (e_{ii} - a_i^2) = \text{Tr}(e) - \| e \|^2$$  \hspace{1cm} (2)

where $\| x \|$ represents the sum of all elements of matrix $x$.

The modularity measure is used like a stop criteria for the algorithm. It is worth noting here that modularity values close to 1 (maximum value) indicate strong community structure.

III. Examples of Communities Detection

To summarize our contribution in this paper we give some examples. Firstly a simple but very clear example is illustrated in the Figure 6(a) it has been assigned equal weight to all edges which means that collaborators who worked together published six articles in all cases.

Figure 6(b) presents the same graph as a multi-weighted graphs, the six publications have been distributed to different priority levels. In this example has been considered three priority levels which can be seen following the format: [first_priority_level, second_priority_level, third_priority_level].

The modified algorithm was applied in this example. Figure 7(a) shows the case in which all weights have the same priority level equal to 1. In this case the algorithm has been guided by the natural division of the network and it found 5 communities. This first result is the same if we do not use the multi-weighted network. Figures 7(b)-7(d) show how the multi-weighted influences in the detection of communities. When the priority levels are modified, the vertex $a$ moves of community because of the real weight of the edges exchange. The vertex $a$ it is associated to the community with which has greater strength.

For the next example, the network shown in Figure 8(a) was taken as input, and the Figure 8(c) shows the communities detected after applying the algorithm without priority levels and only with single values as weights. The algorithm only detected two communities and with the next removal the modularity decrease so the algorithm detect only two communities.

In the other hand when the algorithm took the network shown in Figure 8(b) and works with priority levels and multi-weighted edges returns the communities shown in Figure 8(d). In this case the algorithm detected three communities within which the weight of edges are high. The value of modularity was higher in the case that was used multi-weighted graph and multi priority levels.

This final example was made taking as input the real collaboration graph of Department of Computer Science - IME - USP. 41 Lattes curricula of scientist were considered to generate the collaborations graph shown in Figure 9(a) which is based on their publications.

The real co-authorships network was extracted by scriptLattes [8] which manage real information about bibliographical productions of Brazilian academic institutions. In our work we considered collaboration graphs generated by scriptLattes with the following initial conditions:

There are three priority levels and each of them has a specific weighting:

- Level 1 (weighting = 3):
  - Articles in scientific journals.
  - Book published/organized.
  - Book chapter published.
Figure 7: Detection of communities with priority levels of: (a) $[1,1,1]$, (b) $[3,2,1]$, (c) $[2,3,1]$, (d) $[1,2,3]$.

Figure 8: (a) weighted network. (b) multi-weighted network. (c) communities detected using single weight in the edges. (d) communities detected working with the multi-weighted graph.

- Level 2 (weighting =2):
  - Complete works published in proceedings of conferences.
- Level 3 (weighting =1):
  - Articles in newspapers/magazines.
  - Expanded summary published in proceedings of conferences.
  - Summary published in proceedings of conference.

Figure 9(b) shows the communities detected by the algorithm, taking as input the collaboration network showed in the Figure 9(a) and working only with a weight in the network’s edges equivalent to the sum of all publications between the authors who are linked by the edge.

In this case, the algorithm (without modification) detected six communities which are represented in different colors, the detection process was guided by the natural community
Figure 9: (a) Collaboration Graph - Department of Computer Science - IME - USP. (b) Communities detected using a single weight equivalent to the sum of all publications between the authors who are linked by the edge. (c) Communities detected using multi-weighted edges and priority levels ([3, 2, 1]).
structure and the constraint that co-authors who have more publications together must be in the same community. The process was completed in twenty six iterations and by the end of this the maximum value of modularity was 0.70.

Figure 9(c) shows that the modified algorithm applied to collaboration graph showed in the Figure 9(a) detected six communities which you can see in different colours. The process took twenty five iterations and reached maximum value for modularity equal 0.72.

Both process, using a single value for the edge’s weight and using multi-weighted and priority levels in edges, returned six communities but different between them. Note that some vertices from communities detected in the first process (see Figure 9(b)) were moved to other community in the second process (see Figure 9(c)), specifically three vertices from community_2 were moved to community_1, this happened because the multi-weighted and priority levels in edges produced a change in the real weight of the edges, then these vertices have a cohesion force stronger for the community_1 than the community_2.

The divisions (9(b) and 10(c)), both were good if we consider that the modularity reached were 0.70 and 0.72 respectively and the second process exceeds the first in modularity value and reduces the number of iterations.

In these experimental results were presented the impact of including multi-weighted and multi-priority levels in edges, this impact includes a reduction in the number of iterations and an increase in the modularity value which demonstrate that our proposal improves the results. The modified algorithm for community detection allows, in addition to natural community structure detection, keep the constraint that co-authors who have more publications together must be in the same community, a more realistic representation and division of networks because considers the importance of publications in the network’s edges.

IV. CONCLUSION

In this paper was introduced different priority level to calculate the real weight for edges which is based in multi-weighted graphs and it has an important impact in the detection of communities within co-authorship networks.

The presented approach give a more realistic representation of co-authorship networks and despite being simple, gives good results. The modularity measures obtained in the tests (between 0.5 and 0.8) show that the divisions in communities are relevant. It is worth noting that this approach uses a divisive technique, i.e., the approach allows removing edges progressively in order to detect communities. The edges to be removed are selected using the algorithm shortest-path betweenness.

REFERENCES


