International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1, October, 2019

Sub-Community Graph Retrieval from a Compressed Community Graph using Graph Mining

Bapuji Rao, Sarojananda Mishra

Abstract: ***Community detection and its retrieval is one of the most relevant and important topics in graph mining. Hence it is treated as one of the important applications in the field of social network analysis. Community detection plays an important role in a large community graph by enabling and selecting the desired community's sub-graph. The proposed algorithm detects and extracts the desired sub-community graph from a compressed community graph for further analysis purpose. The authors present both theoretical and experimental results with three benchmark social networks. The proposed technique is efficient in terms of complexities.***

Keywords: ***adjacency community matrix, community graph, sub-community graph, self-loop or cycle, weights.***

I. INTRODUCTION

In data mining one of the hot research topics is the analysis of complex networks. Community network is one of the examples of social network. A community can also be considered as cluster, is referred to as vertices with a high density of edges among the graph [11]. The detected community provides reliable information of the structural properties of the community network [4, 11], the interactions information among the agents of a network [3] or the role of the agents to develop inside the network [16]. Detection of disjoint communities in undirected and un-weighted networks by maximizing WCC (is a metric based on triangle structures in a community) by using the method Scalable Community Detection (SCD) and also proposed community metric [17]. A community or a module or a cluster can be considered as a group of nodes with more connectivity amongst its members than between its members [11]. Such groups of nodes or communities are said to be as one organizational unit in social networks [2, 5, 15]. The authors propose an efficient method of community detection and its retrieval of sub-community graph from a compressed community graph. It is an extended work of "Algorithm for Retrieval of Sub-Community Graph from a Compressed Community Graph Using Graph Mining Technique" [13]. The goal of this work is to detect sub-community graph by expanding the compressed community graph. The proposed has been implemented by using three benchmark community networks viz. Dolphins Network [10], Zachary's Karate Club [19], and Football Team Graph [7].

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Bapuji Rao*, CSE, Biju Patnaik University of Technology, Rourkela, India. Email: bapuji.research@gmail.com
Sarojananda Mishra, CSEA, Indira Gandhi Institute of Technology, Sarang, India. Email: sarose.mishra@gmail.com

II. LITERATURE SURVEY

The authors observe different community detection algorithms with different strategies in the literature. So, one of the community detection algorithms is maximizing modularity [7]. The algorithm reports that modularity has resolution limits [1, 6]. In a large graph, the modularity technique is unable to detect small and well defined communities. However a tree-like structure is created by maximization, which is unable to consider as communities. Due to the above reason a multilevel approach has been proposed which is able to construct graphs with hundreds of millions of objects [3]. The created graph quality of results decreases when size of the graph is considerably increases [8].

An important tool called Random walk which is the combination of several community detection algorithms in one place. Random walk strategy has been used in Walktrap [12]. Another community detection algorithm is Infomap [14] which adapts random walks technique. According to the comparison of community detection algorithms performed by Lancichinetti *et al*. [8], Infomap algorithm is considered as the best community detection algorithms. The algorithm BigClam is proposed by [18] which are based on computing an affiliation of vertices to communities that maximizes an objective function using non negative matrix factorization.

III. PROPOSED ALGORITHM

A. Algorithm for Extraction of Sub-Community Graph

Algorithm ExCompComGra( ) [Algorithm convention [9]]

Cnum: Community number of community sub-graph to extract.

nc: Number of communities.

TNC: To store total number of community members.

CNTA[nc][3]: Matrix to hold community numbers, total community members, and Actual edges of community.

CCMatrix[nc+1][nc+1]: Compressed Community Adjacency Matrix of order (nc+1)X(nc+1).
EXAM[TNC+1][TNC+1]: Expanded adjacency matrix of order (TNC+1)X(TNC+1).
"CommData.Txt": Data file to contain total number of communities, community code, and number of members in each community.
"TotEdge.Txt": Data file to contain total number of edges in the same community as well as dissimilar edges among the communities.
"EdgeData.Txt": Data file to contain the edge details of the community members i.e. "From Community Member" and "To Community Member".
"Eadjmat.Txt": Data file to write the expanded compressed community adjacency matrix.

```
{
  open("CommData.txt"); // to open the file
  read(nc); // to read number of communities
 i:=1;
while(Not EOF())
{
  read(CNTA[i][1]);
  read(CNTA[i][2]);
  i:=i+1;
 }
close("CommData.Txt");  // to close the file
// to assign the community codes
  for i:=2 to (nc+1) do
  {
   CCMatrix[1][i] := CNTA[i-1][1];   // in first row
   CCMatrix[i][1] := CNTA[i-1][1];   // in first column
   }
    open("TotEdge.Txt");   // to open the file
for i:=1 to to (nc+1) do
  for j:=1 to (nc+1) do
     read(CCMatrix[i+1][j+1]);    // to read ncXnc data
close("TotEdge.txt");
// to display Compressed Community Adjacency Matrix
 for i:=1 to (nc+1) do
  for j:=1 to (nc+1) do
   display(CCMatrix[i][j]);
// to calculate number of similar community edges
for i:=2 to (nc+1) do
  CNTA[i-1][3] := CCMatrix[i][i];
EdgeVerification( );  // procedure for edge verification
TNC:=0;
   for i:=1 to n do
     TNC := TNC + CNTA[i][2];
Index:=2;   code:=0;   mcode:=0;
for i:=1 to nc do  // to assign Community Member Codes
{
  code := 10*CNTA[i][1];
  for j:=1 to CNTA[i][2] do {
    if(j>=1 and j<=9) mcode := code + j;
    if(j>=10 and j<=99) mcode := (code*10) + j;
    EXAM[1][Index] := mcode;  // row assignment
    EXAM[Index][1] := mcode;  // column assignment
    Index := Index+1;
  }
 }
open("EdgeData.Txt"); // to open the file
while(Not EOF())
{
  read(rc); // to read "From Community Member Code"
  read(cc);   // to read "To Community Member Code"
for i:=2 to (TNC+1) do
   if (EXAM[i][1]=rc) break;   // to detect row-side
for j:=2 to (TNC+1) do
   if (EXAM[1][j]=cc) break;  // to detect column-side
// to assign edge value 1 on row-side and column-side
EXAM[i][j] := EXAM[j][i] := 1;
}
close("EdgeData.Txt");  // to close the file
//  to open the file for writing EXAM[ ][ ] matrix
open("Eadjmat.Txt");
   for i:=1 to (TNC+1) do
     for j:=1 to (TNC+1) do
       write(EXAM[i][j]);
close("Eadjmat.Txt");   // to close the file
read(Cnum); // to read community number from the user
// procedure to extract Cnum sub-community matrix
Extraction (Cnum);
}
```

B. Procedure for Edge Verification

Procedure EdgeVerification( )
rc, cc: To assign "from node code" and "to node code" from dataset file.
code: To assign the community code.
count: To count number of edges.
csum: To assign the total edges.

```
{
  open("EdgeData.Txt"); // to open the file
   j:=1;
   count:=0;
  while(Not EOF())
  {
     read(rc);    // to read "from community code"
     read(cc);   // to read "to community code"
     if(rc>=10 and rc<=99) code:=rc/10;
     if(rc>=100 and rc<=999) code:=rc/100;
  if(code=j) count:=count+1;
 else
 {
  csum:=0;
  for i:=j to (nc+1) do
     if(i=j) csum:=csum+CCMatrix[j][i]/2;
       else csum:=csum+CCMatrix[j][i];
if(count≠csum) {display("Edges not Matched"); exit; }
        j:=j+1;
        count:=1;
    }
 }
   csum:=0;
   for i:=j to (nc+1) do
     if(i=j) csum:=csum+CCMatrix[j][i]/2;
       else csum:=csum+CCMatrix[j][i];
 // to check the last node's edges
if(count≠csum) {display("Edges not Matched"); exit;}
close("EdgeData.Txt");  // to close the file
}
```

C. Procedure for Extraction of Sub-Community Adjacency Matrix

Procedure Extraction (Cnum)

Cnum: Community number for detection and extraction as sub-community adjacency matrix.

Low, Up: To assign Lower Bound Index and Upper Bound Index of sub-community adjacency matrix.

code: To assign the community member code.

```
{
 Flag := 0;
 for i:=2 to (TNC+1) do
 {
  if(Cnum>=1 and Cnum<=9)
  {
    if(EXAM[1][i]>=10 and EXAM[1][i]<=99)
                          code := EXAM[1][i]/10;
    if(EXAM[1][i]>=100 and EXAM[1][i]<=999)
                          code := EXAM[1][i]/100;
  }
 if(Cnum>=10 and Cnum<=99)
 {
    if(EXAM[1][i]>=100 and EXAM[1][i]<=999)
                         code := EXAM[1][i]/10;
    if(EXAM[1][i]>=1000 and EXAM[1][i]<=9999)
                         code := EXAM[1][i]/100;
  }
  if (Cnum = code)
  {
   if (Flag = 0) {  Flag := 1;   Low := i; }
   Up := i;
  }
 }
 // to extract Cnum Sub-Community adjacency matrix
 if (Flag = 1)
 {
  for i:=Low to Up do
  {
   display(EXAM[i][1]);
      for j:=Low to Up do
         display(EXAM[i][j]);
  }
 }
 else display("not found");
}
```

The algorithm, ExCompComGra( ) has two procedures namely EdgeVerification( ) and Extraction(Cnum). The algorithm opens the 1st dataset file, "CommData.Txt" which contains total numbers of communities, unique community codes, number of members in each community, and the community members' code of the compressed community graph. These data are read from the dataset file and assigned to the matrix, CNTA[nc][3].

Then the algorithm opens the 2nd dataset file, "TotEdge.Txt" which contains the compressed community graph's edge details i.e., similar edges as well as dissimilar edges of all the nc-communities. These details are read and assigned to the matrix, CCMatrix[nc+1][nc+1]. The similar edges of nc-communities are collected diagonally from the matrix, CCMatrix[nc+1][nc+1] and assigned to the 3rd column of the matrix, CNTA[nc][3].

The 1st procedure, EdgeVerification( ) which verifies the total edges (i.e. both similar and dissimilar edges) available in the matrix, CCMatrix[nc+1][nc+1] and the 3rd dataset file, "EdgeData.Txt". If the total edges from both the sources are not same, then the procedure stops by displaying a proper message.

Then the algorithm generates the community members' code by using the 1st column and the 2nd column of the matrix, CNTA[nc][3]. These members' code are assigned at the 1st row and 1st column of the matrix, EXAM[TNC+1][TNC+1]. Then the algorithm opens the 3rd dataset file, "EdgeData.Txt" which contains the edge details of the expanded community graph and these details are assigned to the matrix, EXAM[TNC+1][TNC+1]. Finally, the algorithm opens another file, "EadjMat.Txt" for writing the expanded adjacency matrix, EXAM[TNC+1][TNC+1].

Finally, the algorithm calls the 2nd procedure, Extraction (Cnum) for extraction of Cnum sub-adjacency matrix from the expanded adjacency matrix, EXAM[TNC+1][TNC+1]. If Cnum is found in the expanded adjacency matrix, EXAM[TNC+1][TNC+1], then the procedure successfully retrieves the sub-community adjacency matrix of Cnum. Otherwise it displays a proper message. The time complexity of the extended algorithm is **O(n²)**.

IV. EXPERIMENTAL RESULTS

D. Example-I (Zachary's Karate Club)

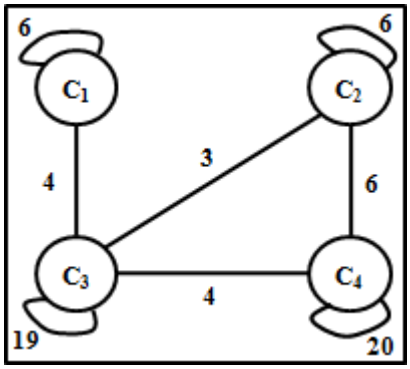

Fig. 1.Compressed Karate community graph.

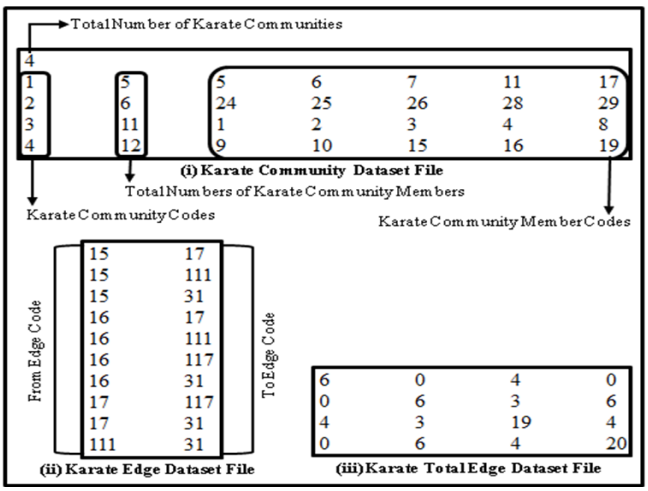

Fig. 2.Compressed Karate community graph datasets.

The karate club has 34 members at a US university, described by Wayne Zachary [19].

These 34 members have been divided into four clubs C_1, C_2, C_3, and C_4 whose member codes are assigned from 1 to 34. Hence the compressed karate community graph has four clubs or communities C_1, C_2, C_3, and C_4; which has total numbers of similar edges 6, 6, 19, and 20 respectively.

These similar edges are the total edges among its members and considered as self-loop or cycle which is the weight of the club or community C_1. Similarly, the total numbers of dissimilar edges (i.e. weights) from C_1 to C_3 is 4 (i.e. total edges between the members of club or communities C_1 and C_3), C_2 to C_3 is 3, C_2 to C_4 is 6, and C_3 to C_4 is 4 respectively and depicted in "Fig. 1".

The authors have created three datasets and depicted in "Fig. 2".The 1st dataset contains the total numbers of clubs or communities, the codes of clubs or communities, the total numbers of club or community members, and the members' respective code and depicted in "Fig. 2(i)".

The 2nd dataset contains edge details of karate community members and depicted in "Fig. 2(ii)". Similarly, the 3rd dataset contains the total edge details (i.e. similar edges and dissimilar edges) of compressed karate community graph and depicted in "Fig. 2(iii)". The diagonal data represents the total numbers of similar edges of the communities C_1, C_2, C_3, and C_4 respectively.

E. Result-I

KCode	15	16	17	111	117	224	225	226
15	0	0	1	1	0	0	0	0
16	0	0	1	1	1	0	0	0
17	1	1	0	0	1	0	0	0
111	1	1	0	0	0	0	0	0
117	0	1	1	0	0	0	0	0
224	0	0	0	0	0	0	0	1
225	0	0	0	0	0	0	0	1
226	0	0	0	0	0	1	1	0

Fig. 3.Karate community adjacency matrix.

Enter Karate Team Community Number From [1 to 4] : 4

Karate Team Community-4's Sub-Adjacency Matrix

KtCC4	49	410	415	416	419	421	423	427	430	431	433	434
49	0	0	0	0	0	0	0	0	0	1	1	1
410	0	0	0	0	0	0	0	0	0	0	0	1
415	0	0	0	0	0	0	0	0	0	0	0	1
416	0	0	0	0	0	0	0	0	0	0	1	1
419	0	0	0	0	0	0	0	0	0	0	1	1
421	0	0	0	0	0	0	0	0	0	0	1	1
423	0	0	0	0	0	0	0	0	0	0	1	1
427	0	0	0	0	0	0	0	0	1	0	0	1
430	0	0	0	0	0	0	0	1	0	0	1	1
431	1	0	0	0	0	0	0	0	0	0	1	1
433	1	0	0	1	1	1	1	0	1	1	0	1
434	1	1	1	1	1	1	1	1	1	1	1	0

Fig. 4.Karate community-4's sub-adjacency matrix.

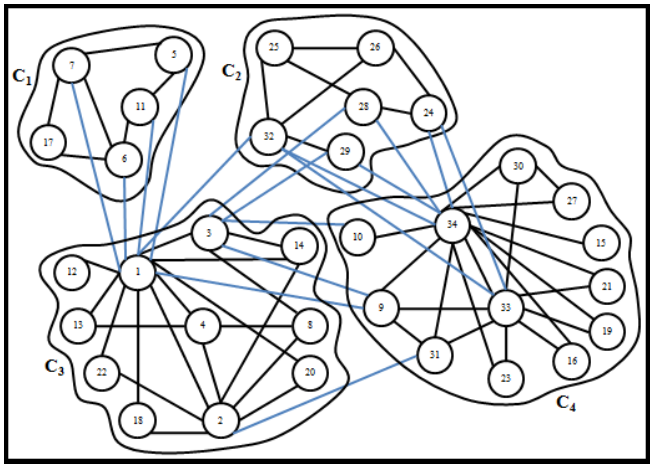

Fig. 5.Expanded Karate community graph.

The expanded karate community adjacency matrix is depicted in "Fig. 3". The community-4's sub-adjacency matrix has been extracted successfully and depicted in "Fig. 4". Finally, the authors have drawn the expanded karate community graph from "Fig. 3" and depicted in "Fig. 5". The blue color edges are the indication of dissimilar edges between the dissimilar community's members.

F. Example-II (Dolphin Network)

The dolphin social network has 62 dolphins and there is a frequent association among them, compiled by Lusseau *et al.* [10]. The 62 dolphins are divided into four communities C_1, C_2, C_3, and C_4. The dolphins' codes are assigned from 1 to 62. The communities C_1, C_2, C_3, and C_4 have total numbers of similar edges 36, 10, 26, and 36 respectively. These similar edges are considered as self-loop or cycle which is the weight of the respected community. Similarly, the total numbers of dissimilar edges (i.e. weights) from C_1 to C_2 is 3 (i.e. total edges between the members of communities C_1 and C_2), C_1 to C_4 is 4, C_2 to C_3 is 3, C_2 to C_4 is 4, and C_3 to C_4 is 13 respectively and depicted in "Fig. 6".

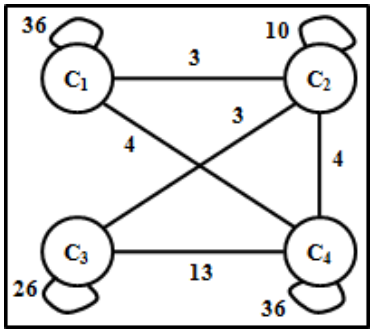

Fig. 6.Compressed Dolphin community graph.

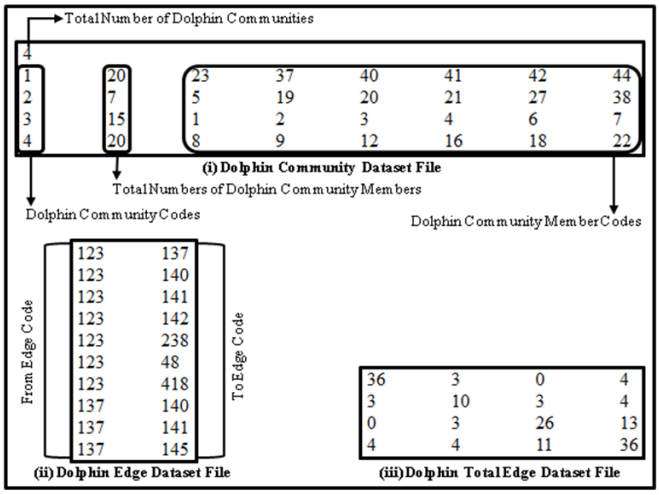

Fig. 7.Compressed Dolphin community graph datasets.

The authors have created three datasets for the compressed dolphin community graph depicted in "Figure 7". The 1st dataset contains the total numbers of communities, the unique codes of communities, the total numbers of dolphin members, and the dolphin members' respective code and depicted in "Fig. 7(i)".The 2nd dataset contains the edge details of the dolphin community members and depicted in "Fig. 7(ii)". Similarly, the 3rd dataset contains the total edge details (both similar and dissimilar edges) of the compressed dolphin community graph and depicted in "Fig. 7(iii)".

The diagonal data represents the total numbers of similar edges related to the communities C_1, C_2, C_3, and C_4 respectively.

G. Result-II

The expanded dolphin community adjacency matrix is depicted in "Fig. 8". The community-2's sub-adjacency matrix has been retrieved successfully and depicted in "Fig. 9". The authors have drawn the expanded dolphin community graph from "Fig. 8" and depicted in "Fig. 10". The edges in blue color are the indication of dissimilar edges between the dissimilar community's members.

DCode	123	137	140	141	142	144	145
123	0	1	1	1	1	0	0
137	1	0	1	1	0	0	1
140	1	1	0	0	1	1	1
141	1	1	0	0	0	0	0
142	1	0	1	0	0	0	0
144	0	0	1	0	0	0	1
145	0	1	1	0	0	1	0

Fig. 8.Dolphin community adjacency matrix.

Dolphin Community-2's Sub-Adjacency Matrix

DnCC2	25	219	220	221	227	238	239
25	0	1	0	1	0	0	0
219	1	0	1	1	1	1	0
220	0	1	0	0	0	1	1
221	1	1	0	0	0	0	1
227	0	1	0	0	0	1	0
238	0	1	1	0	1	0	0
239	0	0	1	1	0	0	0

Fig. 9.Dolphin community-2's sub-adjacency matrix.

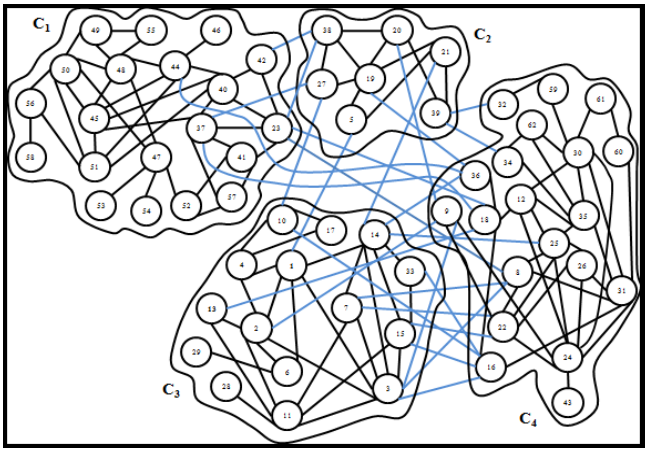

Fig. 10. Expanded Dolphin community graph.

H. Example-III (Football Team Graph)

The network of American football games between Division IA Colleges has 115 numbers of football teams, compiled by Girvan & Newman, 2002[7]. The authors have considered the above benchmark example of football team community graph and divided 115 football teams into twelve clubs or communities C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}, C_{11}, and C_{12} respectively. These clubs or communities have total numbers of similar edges 36, 26, 44, 48, 31, 1, 50, 28, 40, 48, 10, and 30 respectively. These similar edges of clubs or communities are considered as self-loop or cycle which is the weight of the respected community and depicted in "Fig. 11". Similarly, the total numbers of dissimilar edges (i.e. weights) from C_1 to C_2 is 5(i.e. total edges between the members of communities C_1 and C_2), C_1 to C_3 is 2, C_1 to C_4 is 1, and so on.

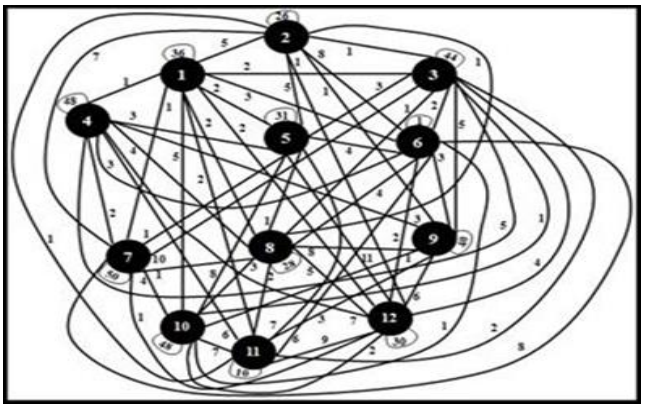

Fig. 11. Compressed Football community graph.

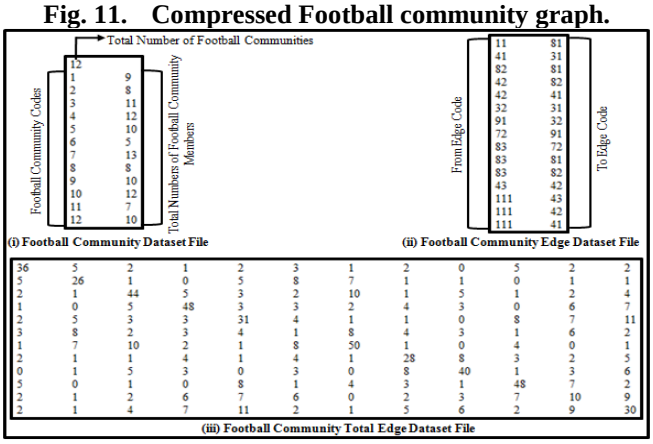

Fig. 12. Compressed Football community graph datasets.

The authors have created three datasets for the compressed football community graph and depicted in "Fig. 12". The 1st dataset contains total numbers of clubs or communities, the unique codes of clubs or communities, and the total number of football teams present in each club or community. Here the football teams of each club or community is numbered from 1 to n depicted in "Fig. 12(i)". The 2nd dataset contains the edge details of football teams and depicted in "Fig. 12(ii)". Similarly, the 3rd dataset contains the total edge details (i.e. similar and dissimilar edges of football teams) of the compressed football community graph. The diagonal data represents the total numbers of similar edges of the clubs or communities C_1 to C_{12} depicted in "Fig. 12(iii)".

I. Result-III

The expanded football community adjacency matrix is depicted in "Fig. 13". The authors have successfully retrieved the football community-7's sub-adjacency matrix depicted in "Fig. 14". Finally, the authors have drawn the expanded football community graph from "Fig. 13" and depicted in "Fig. 15".

FCode	11	12	13	14	15	16	17	18	19	21	22
11	0	1	1	1	1	1	1	1	1	0	0
12	1	0	1	1	1	1	1	1	1	0	0
13	1	1	0	1	1	1	1	1	1	1	0
14	1	1	1	0	1	1	1	1	1	0	0
15	1	1	1	1	0	1	1	1	1	0	0
16	1	1	1	1	1	0	1	1	1	0	0
17	1	1	1	1	1	1	0	1	1	0	0
18	1	1	1	1	1	1	1	0	1	0	0
19	1	1	1	1	1	1	1	1	0	0	0
21	0	0	1	0	0	0	0	0	0	0	1
22	0	0	0	0	0	0	0	0	0	1	0

Fig. 13. Football community adjacency matrix

Enter Football Team Community Number From [1 to 12] : 7

Football Team Community-7's Sub-Adjacency Matrix

FtCC7	71	72	73	74	75	76	77	78	79	710	711	712	713
71	0	1	1	1	0	1	1	1	0	0	0	1	0
72	1	0	1	1	1	1	1	1	1	1	1	1	1
73	1	1	0	1	1	1	1	0	1	1	1	0	1
74	1	1	1	0	1	1	1	1	1	1	0	1	0
75	0	1	1	1	0	1	1	1	1	1	1	1	1
76	1	1	1	1	1	0	0	1	1	1	1	0	1
77	1	1	1	1	1	0	0	1	1	1	1	1	0
78	1	1	0	1	1	1	1	0	1	1	0	1	0
79	0	1	1	1	1	1	1	1	0	1	1	0	1
710	0	1	1	1	1	1	1	1	1	0	1	0	1
711	0	1	1	0	1	1	1	0	1	1	0	0	1
712	1	1	0	1	1	0	1	1	0	0	0	0	1
713	0	1	1	0	1	1	0	0	1	1	1	1	0

Fig. 14. Football community-7's sub-adjacency matrix.

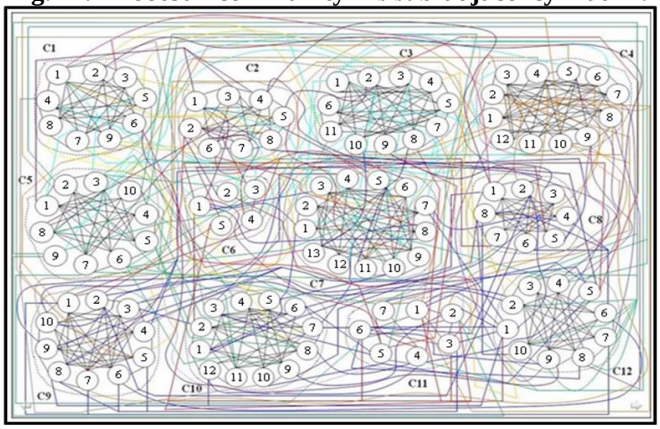

Fig. 15. Expanded Football community graph.

The algorithm was implemented in TurboC++ programming language. The experiment was run on Intel Core I5-3230M CPU + 2.60 GHz Laptop with 4GB memory running MS-Windows 7.

V. CONCLUSION

This paper is an extended work for detection and retrieval of a particular sub-community graph from a compressed community graph. The earlier literature findings, example and algorithm are available in the article [13]. The proposed algorithm has successfully extracted three benchmark community graphs' sub-community graph from their respective compressed community graphs. The overall results have been found satisfactory.

REFERENCES

1. Bagrow J. P., *Are communities just bottlenecks? Trees and treelike networks have high modularity*, CoRR, abs/1201.0745, 2012.
2. Barab´asi A.-L., and Oltvai Z. N., "Network biology: understanding the cell's functional organization", Nature Reviews Genetics, (2):101–113, 2004.
3. Blondel V. D., Guillaume J-L., Lambiotte R., and Lefebvre E., "Fast unfolding of communities in large networks", *JSTAT*, 2008:P10008, 2008.
4. Boccaletti S., Latora V., Moreno Y., Chavez M., and Hwang D.-U., "Complex networks: Structure and dynamics", *Physics reports*, 424(4):175–308, 2006.
5. Feld S. L., "The focused organization of social ties", *American J. of Sociology*, 86(5):1015–1035, 1981.
6. Fortunato S., and Barth´elemy M., "Resolution limit in community detection", *PNAS*, 104(1):36, 2007.
7. Girvan M., and Newman M. E. J., "Community structure in social and biological networks", *Proceedings of the National Academy of Sciences of the United States of America*, 99(12), 7821–7826, 2002. DOI:10.1073/ pnas.122653799 PMID:12060727
8. Lancichinetti A., "Community detection algorithms: a comparative analysis", *Phy. Rev. E*, 80(5):056117, 2009.
9. Lipschutz S., *Schaum's outline of Data Structures*, Tata McGraw-Hill Publishing Company Limited, 7 West Patel Nagar, New Delhi 110 008.
10. Lusseau D., Schneider K., Boisseau O. J., Haase P., Slooten E., and Dawson S. M., "The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations", *Behavioral Ecology and Sociobiology* 54, 396-405, 2003.
11. Newman M., and Girvan M., "Finding and evaluating community structure in networks", *Phy. Rev. E*, 69(2):026113, 2004.
12. Pons P., and Latapy M., "Computing communities in large networks using random walks", *J. Graph Algorithms Appl.*, 10(2):191–218, 2006.
13. Rao B., Mitra A., and Mondal J., "Algorithm for Retrieval of Sub-Community Graph from a Compressed Community Graph Using Graph Mining Technique", *Elsevier Procedia Computer Science*, 57(2015), pp. 678–685, 2015. DOI: 10.1016/j.procs.2015.07.444
14. Rosvall M., and Bergstrom C., "Maps of random walks on complex networks reveal community structure", *PNAS*, 105(4):1118, 2008.
15. Simmel G., *Conflict and the web of group affiliations*, Simon and Schuster, 1964.
16. Wang Y., Cong G., Song G., and Xie K., "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks", In SIGKDD, pages 1039–1048. *ACM*, 2010.
17. WWW'14, April 7–11, Seoul, Korea, 2014.
18. Yang J., and Leskovec J., "Overlapping community detection at scale: a nonnegative matrix factorization approach", In *WSDM*, pages 587–596, 2013.
19. Zachary W. W., "An information flow model for conflict and fission in small groups", *Journal of Anthropological Research* 33, 452-473, 1977.

AUTHORS PROFILE

Bapuji Rao is currently pursuing PhD (CSE) from Biju Patnaik University of Technology (BPUT), Rourkela, Odisha, India. He received M.Tech (Computer Science) from Berhampur University, Berhampur, Odisha, India. He has published seventeen international conference papers, one national conference paper, five chapters and a research text book entitled "Representation of Call-Duration and Social Graph as Multi-Layer Graphs – A Graph Mining Techniques", LAP LAMBERT Academic Publishing, Mauritius. His Current Research area focuses on Graph Mining, Social Network, Data Mining, Opinion Mining, Attributed Graph, and Multi-Layer Graph.

Prof. (Dr.) Sarojananda Mishra is currently working as Professor and Head of the Department of Computer Science, Engg. & Applications at Indira Gandhi Institute of Technology (IGIT), Sarang, Dhenkanal, Odisha, India. He has published more than 80 papers in International Journals and National Journals of repute. His research area focuses on fractal graphics, fractal geometry and internet data analysis. He has more than 25 years of teaching and research experiences. Many students obtained PhD degree and are continuing their PhD and M. Tech research work under his guidance.