2016 IEEE/WIC/ACM International Conference on Web Intelligence

Core Periphery Structures in Weighted Graphs using Greedy Growth

Divya Sardana, Raj Bhatnagar
University of Cincinnati
email: sardanda@mail.uc.edu, bhatnark@ucmail.uc.edu

***Abstract*—Core periphery structure is a meso-scale property of complex networks. Core periphery structures can help identify the relationships between cohesive core clusters surrounded by sparse peripheries. The knowledge about such relationships can have many practical applications in real world complex networks. For example, in a web based network between all blogs on different topics, peripheries connecting popular groups could help in the study of flow of information across the web. In this paper, we propose a construction of core periphery structures for weighted graphs. We present a greedy growth based algorithm to extract core periphery structures in weighted graphs. We also score the core periphery associations as a measure of distance between them. Through extensive experimentation using two synthetic and two real world Protein-Protein Interaction (PPI) networks, we demonstrate the usefulness of core periphery structures over simple overlapping clusters obtained by a state of the art clustering algorithm called ClusterONE.**

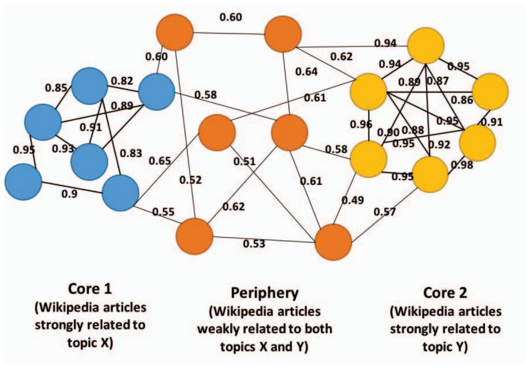

Fig. 1. Illustration of Core Periphery Structures in a weighted graph of Wikipedia articles.

I. Introduction

Community structure is a widely used meso-scale property of complex networks. The analysis of community structures in complex networks can help in a deeper understanding of many real world networks. [1]. Some examples of community structure include groups in social networks and modules in protein-protein interaction (PPI) networks. Overlapping clusters result when nodes have membership in multiple communities. Algorithms finding overlapping clusters can thus be used to discover relationships between network communities.

Core periphery structure is another meso-property of complex networks and can provide a more intuitive understanding of inter cluster relationships across the network. A first formal definition of a core periphery structure was given by Borgatti and Everett as a cohesively connected dense core, loosely connected by a surrounding sparse periphery [2]. It has been demonstrated through experimental results in [3] that core periphery structures arise naturally in networks due to overlapping communities. Core periphery structure analysis can be very useful in many real world networks, such as, web, social, information, food web networks and protein-protein interaction (PPI) networks. In [4], authors perform a core decomposition of the graph to locate influential nodes in web based complex networks like an email communication network (Enron), and a social friendship network from Slashdot. We illustrate the use of core periphery structures in a network of wikipedia articles connected by topic similarity, using an example in figure 1. In figure 1, core 1 corresponds to wikipedia articles strongly relevant to topic X, and core 2 corresponds to wikipedia articles related to topic Y. A periphery connected to both core 1 and core 2 corresponds to articles weakly related to both topics X and Y. This illustrates that core periphery structures can help in studying relationships between groups, thereby helping in the study of flow of information in complex networks. In this paper, we use PPI networks to demonstrate the effectiveness of core periphery structures. Here, core periphery structures could represent a protein complex as a static set of cohesively connected proteins surrounded by a transient set of loosely connected periphery proteins [5].

In this paper, we build a greedy growth based algorithm called ClusterONE-CP to find core periphery structures in a weighted network. We build this algorithm on top of an existing overlapping graph clustering algorithm called ClusterONE and present the consequent core periphery structures obtained. We define a core periphery structure ($C \cup P$) in a weighted graph as an organization of core cluster C and periphery cluster P, such that

i) Nodes $\in$ C are cohesively connected to each other. We call this measure of density as structure based density.
ii) Nodes $\in$ P are loosely connected to the nodes $\in$ C.

The core periphery structures are scored based upon how many standard deviations away is the mean of periphery edges from the core edges. The core periphery structures are further divided into two types: *Type 1* and *Type 2*, based upon their closeness to the core. A graph can consist of multiple core periphery structures. Using two synthetic and two weighted protein-protein interaction networks, we illustrate the significance of dense cores and peripheries found by ClusterONE-CP, than the simple overlapping clusters found by ClusterONE.

978-1-5090-4470-2 2016
U.S. Government Work Not Protected by U.S. Copyright
DOI 10.1109/WI.2016.12

II. Related Work

Traditionally, overlapping clustering has been used to find out relationships among clusters in the network. Several overlapping clustering algorithms have been developed in the research community [7], [8], [9], [10], [11], [12], [6]. However, as demonstrated in [3], most of the traditional overlapping clustering algorithms lead to a consequence that the nodes in community overlaps could be less dense than the non overlapping nodes in the same community.

Core periphery structure represents another model for studying inter cluster relationships. Borgatti and Everett were the first to formalize a model for core periphery structures in a network [2]. According to their model, a core periphery structure exists in a network if it contains a core set in which the nodes are cohesively connected to each other, and a periphery set with nodes loosely connected to the core. Their algorithm divides the complete network into a single core and periphery. The same authors expanded their model of core periphery structures to include multiple cores and peripheries in the network by allowing some low density connections among the periphery nodes [13]. Boyd et al. [14] proposed a Kernighan-Lin algorithm based methodology to partition a social network into core and periphery structures. An enhanced version of their algorithm was developed by Luo et al. [5] to identify k-plex cores in protein protein interaction networks. Rossa et al. [15] used a random walk based algorithm to assign coreness value to nodes in a network. Recently, Yang and Leskovec [3] presented a model of structural communities in a network using overlapping tiles. They use this model to find overlapping clusters and core periphery structures. A graph modification based approach to find core periphery structures for PPI networks has been presented in [16].

III. Methodology

We build a greedy growth based algorithm for finding core periphery structures in weighted graphs. It builds on top of a graph clustering algorithm called ClusterONE [6], hence we name it as ClusterONE-CP. Unlike clusterONE, which only uses a measure of structural density called cohesion for a growing cluster, we also make use of edge weight density in a weighted graph in the form of mean of edges to build peripheries around cores. The cohesion score for a cluster C is defined as follows.

$$Cohesion(C) = \left(\frac{\sum(w_{in}(C))}{\sum(w_{in}(C)) + \sum(w_{out}(C)) + p \times |C|} \right) \tag{1}$$

Here w_{in} denotes the internal edge weights $\in$ C and w_{out} denotes the weight of outgoing edges from C. $p * |C|$ is a penalty term used to model the existence of yet undiscovered edges. Further, a user defined parameter called $MEAN\_OFFSET$ is defined to ascertain the difference in edge weight density of prospective core clusters and their surrounding periphery clusters. A value of 0.2 is used as a default value for $MEAN\_OFFSET$. The algorithm operates in four phases as described next.

A. Initialization Phase

The initialization phase begins with ordering the nodes of the graph for cluster initiation in the greedy growth phase. The nodes of the graph are ranked according to their weighted degree. This order is defined as the cluster initiator order.

B. Greedy Growth Phase

In the greedy growth phase, cluster initiators not previously assigned a cluster label are chosen as seeds for cluster growth. The seed nodes are allowed to grow their cluster as long as the structural density measure, Cohesion of the cluster keeps on increasing. The growth process checks for node addition or deletion from the growing cluster, whichever leads to a greater increase in cohesion. A growing seed stops further growth when a locally optimal cohesive group, say L_i has been formed. At this point, L_i is used to extract its boundary nodesets. The growth phase uses boundary nodesets to facilitate the growth of prospective periphery clusters in the boundary of relatively dense prospective core clusters. Next we describe the extraction of boundary nodesets and how they are used in the growth phase in the next two subsections.

1) Extraction of Boundary nodesets: The greedy growth process for a growing seed results into a locally optimal cohesive cluster say L_i. All external nodes of L_i, incident on at least one boundary edge are checked for possible addition to one of the boundary nodesets *inrange*, *low* or *high* as defined next. The value of $MEAN\_OFFSET$ is used to facilitate this process. In figure 2, nodes A, B and C are examples of boundary nodes of type *inrange*, *low* and *high* respectively for the center blue cluster.

i) Boundary Nodeset *inrange*: All external nodes of L_i, which do not satisfy the structural density constraint, cohesion for L_i, but majority of their adjacent edges to L_i are within $MEAN\_OFFSET$ away from mean($w_{in}(L_i)$).

ii) Boundary Nodeset *low*: All external nodes of L_i, which do not satisfy the structural density constraint cohesion for L_i, and majority of their adjacent edges to L_i have less edge weight than (mean($w_{in}(L_i)$) - $MEAN\_OFFSET$).

iii) Boundary Nodeset *high*: All external nodes of L_i, which do not satisfy the structural density constraint cohesion for L_i, and majority of their adjacent edges to L_i have greater edge weight than (mean($w_{in}(L_i)$) + $MEAN\_OFFSET$).

2) Using boundary nodesets to grow peripheries around cores: In the growth phase, if the growing cluster C adds a node which lies in the boundary nodeset *low* of any locally optimal cluster formed previously, say, L_{prev}, then, the growth phase makes sure to exclude the nodes belonging to L_{prev} in consideration for further growth of C. For example, in figure 2, if node B initiates cluster formation, it will bypass the addition of blue nodes to it as B lies in the blue cluster's boundary nodeset *low*. Instead, it will consider the addition of all red neighboring nodes to its cluster. This helps the prospective periphery clusters to grow around any neighboring cores, rather than merging partially with the core.

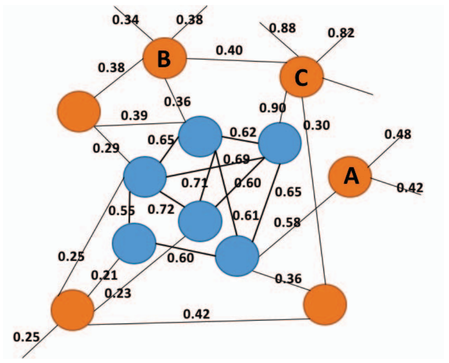

Fig. 2. Example of boundary nodes of type $inrange$ (A), low (B), and $high$ (C) for a core cluster

Algorithms 1 and 2 describe the steps for the growth phase and the steps for extracting boundary nodesets respectively.

C. Overlap Phase

The locally optimal cohesive overlapping clusters formed in the growth phase might have a high overlap, such that merging them will avoid duplicate information. Thus, the merge phase merges highly overlapping pairs of clusters from the growth phase with an overlap score greater than an *overlap threshold* (= 0.5). The overlap score between two clusters C$_i$ and C$_j$ is calculated as

$$OverlapScore(C_i, C_j) = \frac{|C_i \cap C_j|^2}{|C_i| \times |C_j|} \qquad (2)$$

D. Extraction of Core Periphery associations

In the growth phase, cluster initiators are chosen as per their weighted degree. This naturally leads to the formation of structurally denser clusters before the relatively sparser clusters. Therefore, clusters which are formed earlier in the growth phase have a greater chance to become cores than clusters which are formed later in the growth phase. We exploit this property to look for core periphery relationships in the natural cluster formation order. All clusters, O$_i$, are traversed in the order in which they were formed to check for their boundary nodesets *inrange* and *low*. It is to be noted that by now, each node in these boundary nodesets should also have been assigned a cluster label, say, P$_j$, in the growth phase. Core periphery relationships are made between O$_i$ and each cluster P$_j$, if mean(w$_{in}$(O$_i$)) is greater than mean(w$_{in}$(P$_j$)) for *low* boundary clusters, and cohesion(O$_i$) is greater than cohesion(P$_j$) for *inrange* boundary clusters.

Type of Core periphery relationship: Each core periphery relationship (O$_i$, P$_j$) is given a type (1 or 2) based upon the majority nodeset type in the periphery cluster. Type 1 peripheries have majority of nodes belonging to nodeset *inrange*, while type 2 peripheries have majority of the nodes belonging to nodeset *low*.

Score of Core periphery relationship: The core periphery relationship is scored based upon a distance as described below.

Algorithm 1: Greedy Growth Phase

```
Input:
    $SM$: Similarity matrix for graph $G = (V, E)$
    $P$: List of nodes
    $NODE\_PENALTY$: (for calculating cohesion)
Output:
Cluster label set S for each node
Boundary nodesets $inrange, low, high$ for each cluster.
Sort nodelist $P$ in decreasing order of weighted degrees
Initialize cluster label $\theta$
for each node $p_i \in P$ do
    if $S(p_i)$ empty for $p_i$ then
        add $\theta$ to $S(p_i)$
        Cluster nodeset $C_n \leftarrow p_i$
        $cohesion\_bound \leftarrow cohesion(C_n)$
        for
            $p_j \in$ external nodes incident on at least one boundary edge of $C_n$ and $p_i \notin exclude\_set$
        do
            if $cohesion(C_n \cup p_j) > cohesion\_bound$ then
                $chosen\_node \leftarrow p_j$
                $cohesion\_bound \leftarrow cohesion(C_n \cup p_j)$
                $add\_flag \leftarrow true$
                $delete\_flag \leftarrow false$
            end
        end
        for
            $p_j \in$ internal nodes of $C_n$ incident on at least one boundary edge of $C_n$
        do
            if $cohesion(C_n \setminus p_j) > cohesion\_bound$ then
                $chosen\_node \leftarrow p_j$
                $cohesion\_bound \leftarrow cohesion(C_n \setminus p_j)$
                $add\_flag \leftarrow false$ $delete\_flag \leftarrow true$
            end
        end
        if $add\_flag$ is $true$ then
            $C_n \leftarrow C_n \cup chosen\_node$
            For all sets $C'_n$ s.t. $chosen\_node$ is in boundary nodeset $low$ of $C'_n$, add all nodes $\in C'_n$ to $exclude\_set$
        else if $delete\_flag$ is $true$ then
            $C_n \leftarrow C_n \setminus chosen\_node$
            For all sets $C'_n$ s.t. $chosen\_node$ is in boundary nodeset $low$ of $C'_n$, remove all nodes $\in C'_n$ from $exclude\_set$.
        else
            Declare $C_n$ as a locally optimal cluster
            Generate $boundary\_nodesets(C_n)$
    end
end
```

Algorithm 2: Generate Boundary Nodesets

```
Input:
    $C_n$: locally optimal cohesive cluster nodeset
    $SM$: similarity matrix
    $MEAN\_OFFSET$: user defined parameter
Output:
    Boundary nodesets $inrange, low\ and\ high$ for $C_n$
1  for
       $p_j \in$ external nodes of $C_n$ incident on at least one boundary edge
   do
2      $E \leftarrow Set\ of\ edges\ connecting\ p_j\ and\ C_n$
3      $lbound = \mu(C_n) - MEAN\_OFFSET$
4      $ubound = \mu(C_n) + MEAN\_OFFSET$
5      $n_{\text{short}} \leftarrow no.\ edges\ in\ E < lbound$
6      $n_{\text{long}} \leftarrow no.\ edges\ in\ E > ubound$
7      $n_{\text{inrange}} \leftarrow no.\ edges\ in\ E \geq lbound\ and\ \leq ubound$
8      if $n_{inrange} \geq n_{short} + n_{long}$ then
9          boundary nodeset $inrange \leftarrow$ boundary nodeset $inrange \cup \{p_j\}$
10     end
11     else if $n_{short} \geq n_{inrange} + n_{long}$ then
12         boundary nodeset $low \leftarrow$ boundary nodeset $low \cup \{p_j\}$
13     end
14     else
15         boundary nodeset $high \leftarrow$ boundary nodeset $high \cup \{p_j\}$
16     end
17 end
```

Algorithm 3: Extract Core-Periphery Structures

```
Input:
    $S_n$: Set containing cluster nodesets generated after merge phase
Output:
    $C_p$: Data structure containing core-periphery associations $(C, P)$ along with their type and score
1  Initialize $C_P$ as an empty data structure.
2  Store in $O_n$ the ordered merged clusters in $S_n$ by the order in which their constituent clusters were generated in the greedy growth phase
3  for $O_i \in O_n$ do
4      Set $P$ to cluster nodesets to which nodes $\in O_i$'s boundary nodeset $inrange, low$ belong
5      for $P_j \in P$ do
6          if $(P_j \in low\ and\ \mu(O_i) > \mu(P_j))\ or$
           $(P_j \in inrange\ and$
           $cohesion(O_i) > cohesion(P_j))$ then
7              Form $(O_i, P_j)$ as a core-periphery association
8              Set Type$(O_i, P_j)$ as the type of majority of boundary nodes $\in P_j$ (Type 1: $inrange$, Type 2: $low$)
9              Let $E$ be the edgeset connecting $O_i$ and $P_j$
10             CPDistance
               $(O_i, P_j) \leftarrow \frac{|(\mu(edge\ set\ O_i) - \mu(edge\ set\ P_j \cup E))|}{\sigma(edge\ set\ O_i)}$
11             Add $(O_i, P_j)$ along with its type and CPdistance to $C_P$
12         end
13     end
14 end
```

$$CPDistance(O_i, P_j) = \frac{|\mu(w_{in}(O_i)) - \mu(w_{in}(P_j \cup E))|}{\sigma(w_{in}(O_i))} \tag{3}$$

Here E corresponds to the edgeset connecting O$_i$ to P$_j$. Intuitively, this score captures how many standard deviations away is the periphery from the core cluster's mean. Algorithm 3 describes the steps for extracting core periphery associations.

IV. Experimental Evaluation

In this section, we use two synthetic and two real world PPI networks to demonstrate the usefulness of core periphery structures generated by ClusterONE-CP over simple overlapping clusters generated by ClusterONE. For running ClusterONE, we set $nodepenalty$ as 2 and $overlapthreshold$ as 0.5.

A. Results for Synthetic Datasets

We built two synthetic weighted graph datasets 1 and 2. Synthetic dataset 1 has 39 nodes and 74 edges (figure 3). We embedded two strong cores Core 1: (A, B, C, D, E, F, G, H) and Core 2: (AA, AB, AC, AD, AE, AF, AG, AH, AJ, AK, AL) and a weak core Core 3: (R, S, T, U, V) surrounded by peripheries. Synthetic dataset 2 has 49 nodes and 97 edges (figure 6). It was built so as to embed three cohesive cores surrounded by peripheries. The three cores are: Core 1: (A, B, C, D, E, F, G, H) Core 2: (J, K, L, M, N, P, Q) and Core 3: (R, S, T, U, V, W).

1) Comparative analysis of clusters with ClusterONE: Before moving on the description of core periphery associations among the clusters generated by ClusterONE-CP, we present a comparative analysis of the clusters (either core or periphery) obtained by it with those obtained by ClusterONE.

The overlapping clustering results for ClusterONE on synthetic datasets 1 and 2 are presented in figures 3 and 6. The clusters obtained by ClusterONE-CP on synthetic datasets 1 and 2 are demonstrated in figures 4 and 7 respectively. Each cluster has been given its own color. The edges are drawn proportional to edge weight. It is to be noted that clusterONE leaves certain nodes as belonging to no clusters (For example, nodes P and Q in figure 3). However, clusterONE-CP will report such nodes as belonging to peripheries in figure 4. These peripheries can prove to be cruicial connectors between cores, especially for the study of inflormation flow in web based complex networks.

Further, in table I, we present a comparative analysis of

TABLE I
CLUSTERONE VS CLUSTERONE-CP ON SYNTHETIC DATASETS

$Algorithm$	$Dataset$	$nClusters$ $size >$ 2	$Avg.$ $Variance$	$Avg.$ $Struct.$ $Density$
ClusterONE-CP	Synthetic 1	7	0.0061	0.56
ClusterONE	Synthetic 1	6	0.021	0.53
ClusterONE-CP	Synthetic 2	8	0.012	0.53
ClusterONE	Synthetic 2	8	0.018	0.43

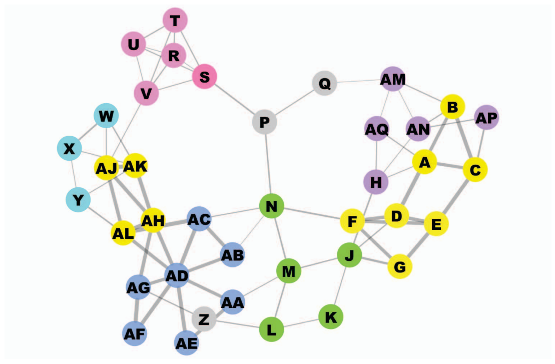

Fig. 3. Overlapping clusters obtained by ClusterONE on Synthetic Dataset 1 (nodes in yellow: overlapping, nodes in gray: no cluster label)

the clusters obtained by ClusterONE and ClusterONE-CP in terms of average structural density and average variance of the generated clusters. We define structural density of a cluster as

$$Strucure\ Density(C) = \left(\frac{|edges(C)|}{|All\ possible\ edges(C)|}\right) \quad (4)$$

Structural density captures how cohesively connected are the clusters as far as structure is concerned. Average variance is used to determine how homogenous are the edge weights inside the cluster. The comparison results in table I demonstrate that the average variance of clusters obtained by ClusterONE-CP is lower than that for ClusterONE for both the synthetic datasets. The obtained average structural density is the same for both the algorithms for synthetic dataset 1 and considerably higher for ClusterONE-CP for synthetic dataset 2. From these results, we can infer that ClusterONE-CP favors clusters with a balance between low variance and a high structural density.

2) Quality of Core Periphery Structures: We present here the core periphery structures obtained by ClusterONE-CP on synthetic datasets 1 and 2 in detail. Tables II and III represent the core periphery associations derived for synthetic dataset

TABLE II
CLUSTER IDS AND CORRESPONDING NODESETS GENERATED BY CLUSTERONE-CP ON SYNTHETIC DATASET 1

$ClusterId$	$Clusternodeset$
0	AL,AG,AE,AD,AH,AF,AA,AB,AC,AJ,AK
1	D,E,F,G,J
2	A,B,C,D,E,F,G
3	N,K,M,L,Z,P,S,Q
4	R,U,T,V,S
5	AP,AQ,H,AM,AN
6	W,X,Y

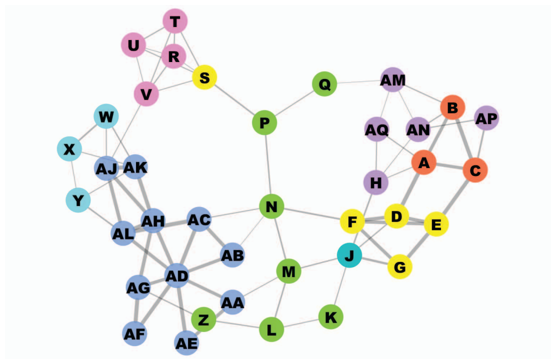

Fig. 4. All Structures obtained by ClusterONE-CP on Synthetic Dataset 1 (nodes in yellow: overlapping)

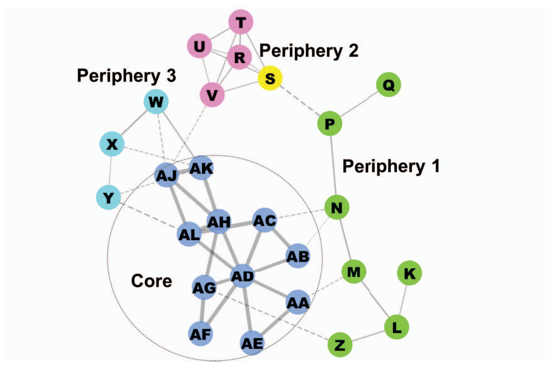

Fig. 5. An example Core Periphery structure obtained by ClusterONE-CP on Synthetic Dataset 1

1. We successfully identify the two dominant cores Cluster 0: (AL, AG, AE, AD, AH, AF, AA, AB, AC, AJ, AK) and Cluster 1:(A, B, C, D, E, F, G) and their peripheries. Table III also lists the type (1 or 2) for each periphery and the corresponding CPdistance as described in section 3. Figure 5 displays an example core and its corresponding peripheries obtained by our algorithm on Synthetic Dataset 1 in a close-up view.

Tables IV and V represent the core periphery associations obtained by ClusterONE-CP for synthetic dataset 2. We identify the three dominant core clusters here as Cluster 0: (A, B, C, D, E, F, G, H), Cluster 1: (R, S, T, U, V, W, CE)

TABLE III
CORE PERIPHERY STRUCTURES BY CLUSTERONE-CP ON SYNTHETIC DATASET 1

$ClusterId$ $(Core)$	$ClusterId$ $(Periphery)$	$Type$	CP $Distance$
0	3	2	10.07
0	4	2	10.9
0	6	2	10.1
1	3	2	2.0
1	5	2	2.5
2	1	2	3.03
2	3	2	9.15
2	5	2	10.55
4	3	1	1.74
3	5	1	2.67

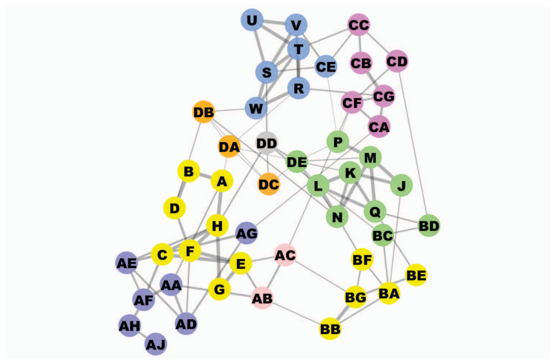

Fig. 6. Overlapping clusters obtained by ClusterONE on Synthetic Dataset 2 (nodes in yellow: overlapping, nodes in gray: no cluster label)

TABLE IV
CLUSTER IDS AND CORRESPONDING NODESETS GENERATED BY OUR ALGORITHM ON SYNTHETIC DATASET 2

$ClusterId$	$Clusternodeset$
0	A,B,C,D,E,F,G,H
1	R,S,T,U,V,W,CE
2	J,K,L,M,N,P,Q,DE
3	AA,AB,AC,AD,AE,AF,AG,AH,AJ
4	CA,CB,CC,CD,CF,CG
5	DC,DD,DE
6	BA,BB,BC,BD,BE,BF,BG
7	DA,DB,DC,DD

and Cluster 2: (J, K, L, M, N, P, DE). Figure 8 displays an example core (R, S, T, U, V, W, CE) and its corresponding peripheries obtained by ClusterONE-CP on Synthetic Dataset 2 in a close-up view. This figure clearly demonstrates the difference in the mean, and structural density of the core and its peripheries. They are in sync with our definition of core periphery structures.

B. Analysis of varying $MEAN\_OFFSET$

ClusterONE-CP uses a user defined parameter called $MEAN\_OFFSET$. This parameter is used to assign boundary nodes of clusters to different boundary nodesets. In table

TABLE V
CORE PERIPHERY STRUCTURES BY OUR ALGORITHM ON SYNTHETIC DATASET 2

$ClusterId$ $(Core)$	$ClusterId$ $(Periphery)$	$Type$	CP $Distance$
0	3	2	4.34
0	5	2	7.17
0	7	2	8.23
1	2	2	1.38
1	4	2	2.45
1	5	2	3.45
1	7	2	4.05
2	4	2	0.75
2	5	1	1.37
2	6	1	0.94
2	7	2	1.65
3	4	2	0.83
3	6	2	1.18
4	6	2	0.26
5	7	1	0

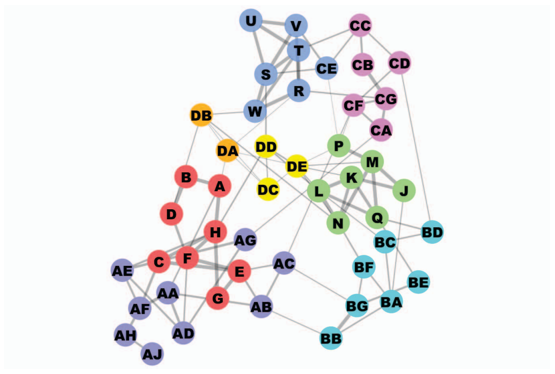

Fig. 7. All Structures obtained by ClusterONE-CP on Synthetic Dataset 2 (nodes in yellow: overlapping)

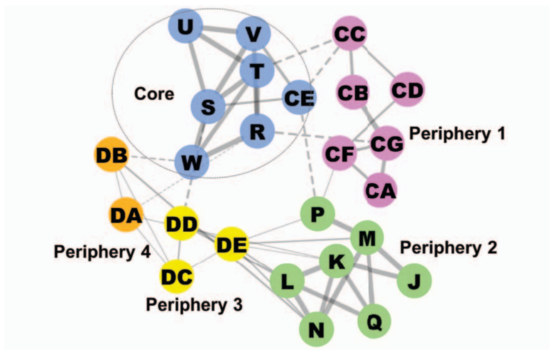

Fig. 8. An example Core Periphery structure obtained by ClusterONE-CP on Synthetic Dataset 2

VI, we present the results of changing $MEAN\_OFFSET$ for synthetic dataset 1. We can see that for higher values of $MEAN\_OFFSET$ like 0.4, the generated clusters start having higher average variance and lower structural density. This deviates from our definition of core periphery structures. Similar trends were obtained on all synthetic and real datasets. Keeping these considerations in mind, we chose a value of 0.2 for $MEAN\_OFFSET$ to obtain a balance of size, low average variance and high structural density for the obtained clusters.

C. Results for Real Datasets

We use two large scale weighted yeast PPI datasets for *Saccharomyces cerevisiae*, namely Gavin [18] and Krogan [19] (table VII), for illustrating the effectiveness of core periphery

TABLE VI
VARIATION OF $MEAN\_OFFSET$ FOR SYNTHETIC DATASET 1

$MEAN$ $OFFSET$	$nClusters$	$nClusters$ $(size > 1)$	$Avg.$ $Variance$	$Avg.$ $Struct.$ $Density$
0.1	7	7	0.006	0.56
0.2	7	7	0.006	0.56
0.3	7	7	0.0063	0.52
0.4	7	7	0.01	0.52
0.5	8	8	0.01	0.52

TABLE VII
REAL PPI DATASETS USED

	Gavin	*Krogan*
No. of proteins	1855	2708
No. of links	7669	7123

TABLE VIII
CLUSTERONE VS CLUSTERONE-CP FOR PPI DATASETS

Algoritm	*nClusters* ($size > 2$)	*Sn*	*PPV*	*Acc.*	*Avg. Mean*
			GavinDataset		
ClusterONE	226	0.56	0.40	0.48	0.39
ClusterONE-CP	245	0.56	0.39	0.47	0.39
			KroganDataset		
ClusterONE	324	0.50	0.42	0.46	0.70
ClusterONE-CP	332	0.49	0.44	0.46	0.73

structures in weighted graphs. Dezso et. al. [20] performed an analysis on experimentally generated *Saccharomyces cerevisiae* yeast protein complexes. They demonstrated that the protein complexes in the yeast are comprised of two regions, namely, core and periphery. The core subunits are highly coexpressed, surrounded by a functionally mixed group of periphery proteins. Based upon this motivation, we present the core periphery results for the two yeast PPI datasets. We use our gold standard as the MIPS catalog of complexes [22] comprising of 203 complexes and a total of 1189 yeast proteins. We present an accuracy comparison for all the clusters obtained by clusterONE vs. clusterONE-CP in table VIII. We use the definition of accuracy as introduced by Brohee and Helden [21]. Accuracy (Acc.) is defined as a geometric mean of clustering-wise sensitivity (Sn) and clustering-wise positive predicted value (PPV). Let T=[t_{ij}] be the confusion matrix of complexes. Let n be the number of predicted clusters and m be the number of gold standard complexes. Let m_i be the number of proteins belonging to gold standard complex i. Sn and PPV are defined as follows.

$$Sn = \left(\frac{\sum_{i=1}^{m} \max\limits_{1 \le j \le n} t_{ij}}{\sum_{i=1}^{m} m_i} \right) \; PPV = \left(\frac{\sum_{j=1}^{n} \max\limits_{1 \le j \le m} t_{ij}}{\sum_{j=1}^{n} \sum_{i=1}^{m} t_{ij}} \right) \quad (5)$$

We observe that the accuracy value for both the datasets is almost the same for both ClusterONE and ClusterONE-CP. This suggests that the core periphery construction doesn't dilute the accuracy of the detected complexes. However, as evident from table IX, a comparison of cores with the corresponding top level clusters from Clusterone shows the significance of cores obtained by Clusterone-CP. The table shows that the average mean and average structural density of cores found by ClusterONE-CP is greater than the corresponding values for top level clusters obtained for both the datasets. Further, as per t-test statistic (alpha=0.05), this improvement in mean and structural density of cores is found to be statistically significant. This highlights that ClusterONE-CP is able to extract the very dense cores which may be a part of relatively bigger sized clusters found by ClusterONE.

Next, we present the properties of cores, coreperipheries (both core and periphery) and peripheries as identified by ClusterONE-CP on the real PPI datasets in table X. It is to be noted that for both the datasets, the cores have a high avg. mean and avg. structural density than coreperipheries which in turn have higher values than peripheries. All types of groups seem to have a low standard deviation. Further, it can be noted that the average essentiality (last column) of cores and coreperipheries is higher than that of peripheries. Here, the essentiality has been calculated using an essential proteins database [23]. It has been observed by researchers that structural characteristics of proteins in a PPI network, like connectivity, could be related to biological properties of proteins, such as essentiality for cell survival [24] [5]. More specifically, it has been found that the number of links per node in a PPI network of *S.Cereviciae* are correlated to the essentiality of such proteins [24]. The results demonstrate that the cores found by ClusterONE-CP get a high average essentiality value than peripheries, thereby adding strength to the results.

In figure 9, we illustrate a subjective example of different types of core periphery structures that we obtain for the Krogan dataset. A core cluster mapped to RSC-complex is identified to have a periphery type-1 as mapped to SWI-SNF-transcription-activator-complex and a periphery type-2 as mapped to Mitochondrial-processing-complex. We searched for literature and found that relationships between these complexes have been studied before by researchers. According to [25], both RSC complex and SWI-SNF-transcription-activator-complex are ATP-dependent chromatin-remodeling complexes

TABLE IX
COMPARISON OF CORES FOUND BY CLUSTERONE-CP WITH CORRESPONDING TOP LEVEL CLUSTERS BY CLUSTERONE FOR PPI DATASETS

	Max. Size	*Avg. Size*	*Min. Size*	*Avg. Mean*	*Avg. Stand. Dev.*	*Avg. Struct. Density*
			GavinDataset			
Cores	39	10	3	0.52	0.15	0.91
Top Level 27 clusters	56	22	8	0.40	0.12	0.57
			KroganDataset			
Cores	8	4	3	0.93	0.06	0.71
Top Level 14 clusters	38	19	5	0.81	0.14	0.31

TABLE X
PROPERTIES OF CORE PERIPHERY STRUCTURES BY CLUSTERONE-CP FOR PPI DATASETS

	No.	*Max. Size*	*Avg. Mean*	*Avg. Stand. Dev.*	*Avg. Struct. Dens.*	*Avg. Essent.*
			GavinDataset			
Cores	27	39	0.52	0.15	0.91	0.37
Core Peripheries	149	59	0.34	0.12	0.57	0.34
Peripheries	52	14	0.32	0.064	0.58	0.16
			KroganDataset			
Cores	14	8	0.93	0.060	0.71	0.23
Core Peripheries	247	37	0.77	0.13	0.45	0.29
Peripheries	55	10	0.48	0.079	0.54	0.22

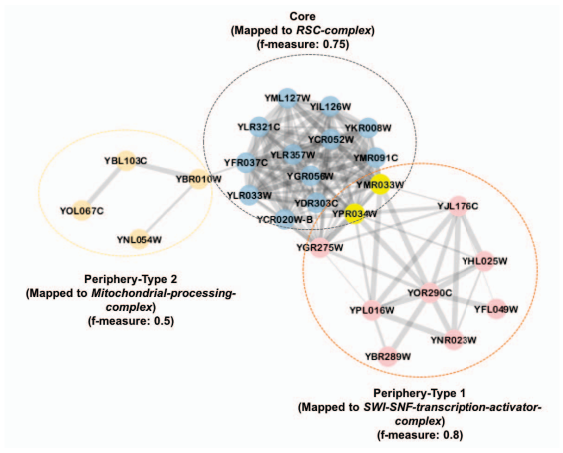

Fig. 9. An example Core-Periphery structure obtained by ClusterONE-CP on Krogan PPI dataset

and both are closely related to each other. However, both regulate different chromatin regions, the former being required for a larger spectrum of genes. This makes RSC more abundant and its function as indispensable for cell survival [26]. Similarly, RSC-complex is identified as a core for Mitochondrial-processing-complex. It has been studied in a very recent paper [27] that RSC-complex is important for mitochondrial function in *Saccharomyces cerevisiae*. Further, figure 9 illustrates our intuition that type-1 peripheries which contain more *inrange* boundary nodes are closer to the core than type-2 peripheries which contain more *low* boundary nodes.

V. Conclusion

In this paper, we developed a greedy growth based algorithm called ClusterONE-CP to find core and periphery structures in weighted graphs. We illustrated the usefulness of core periphery structures over overlapping clusters generated by a state of the art clustering algorithm called ClusterONE. Using two synthetic and two large scale real world PPI networks, we demonstrated that the cores generated by our algorithm have a higher average structural density and a higher average mean than the generated peripheries. For PPI datasets, the cores were even found to have more essential proteins on an average than the peripheries. We further scored the core periphery associations and divided the peripheries in to two types based upon their closeness to the core. We demonstrated the usefulness of the two types of core periphery structures by using a subjective example of protein core periphery structure obtained on one of the real PPI datasets. This illustration of core periphery structures can be seen as a foundational example. In general, core periphery structures found by ClusterONE-CP can be used to gain meaningful insights in other real world complex networks (like web, social and information networks) over simple overlapping clusters found by algorithms like ClusterONE.

References

[1] Lancichinetti, A., Kivelä, M., Saramäki, J. and Fortunato, S.:Characterizing the Community Structure of Complex Networks. In: PLoS ONE, vol. 5, e11976 (2010).
[2] Borgatti, S. P., Everett, M. G.: Models of Core/Periphery structures. In: Social Networks, vol. 21, pp. 375–395 (1999).
[3] Yang, J., Leskovec, J.: Overlapping Communities Explain Core–Periphery Organization of Networks. In: Proceedings of the IEEE 102, vol. 12, pp. 1892-1902 (2014).
[4] Malliaros, F. D., Rossi M. E., Vazirgiannis M.: Locating influential nodes in complex networks. In: Scientific reports 6 (2016).
[5] Luo, F., Li, B., Wan, X., Scheuermann, R.: Core and periphery structures in protein interaction networks. In: BMC Bioinformatics, vol. 10, pp. S8 (2009).
[6] Nepusz, T., Yu, H., Paccanaro, A.: Detecting overlapping protein complexes in protein-protein interaction networks. In: Nature Methods, vol. 9, pp. 471-472 (2012).
[7] Ahn, Y.-Y., Bagrow, J. P., Lehmann, A.: Link communities reveal multiscale complexity in networks. In: Nature, vol. 466, pp. 761–764 (2010).
[8] Airoldi, E. M., Blei D. M, Fienberg, S. E., Xing, E. P.: Mixed membership stochastic blockmodels. In: J. Mach. Learn. Res., vol. 9, pp. 1981–2014 (2007).
[9] Sales-Pardo, M., Guimer'a, R., Moreira, A., Amaral L. A. N.: Extracting the hierarchical organization of complex systems. In: Proc. Nat. Acad. Sci. USA, vol. 104, pp. 18 874–18 874 (2007).
[10] Psorakis, I., Roberts, S., Ebden, M., Sheldon, B.: Overlapping community detection using Bayesian non-negative matrix factorization. In: Phys. Rev. E, vol. 83 (2011).
[11] Palla, G., Derenyi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. In: Nature, vol. 435, no. 7043, pp. 814–818 (2005)
[12] Evans, T. S., Lambiotte, R.: Line graphs, link partitions, and overlapping communities. In: Phys. Rev. E, vol. 80, Art. ID. 016105 (2009).
[13] Everett, M. G., Borgatti, S. P.: Peripheries of cohesive subsets. In: Social Networks, vol. 21, pp. 397 (1999).
[14] Boyd, J. P., Fitzgerald, W. J., Beck R. J.: Computing core/periphery structures and permutation tests for social relations data. In: Social Networks, vol. 28, pp. 166-178 (2006).
[15] Rossa, F.D., Fabio, D., Piccardi C.: Profiling core-periphery network structure by random walkers. In: Sci. Rep. 3, Article no. 1467 (2013).
[16] Bruckner, S., Hüffner, F., Komusiewicz, C.: A graph modification approach for finding core–periphery structures in protein interaction networks. In: Algorithms for Molecular Biology, vol. 10, no. 1, pp.1 (2015).
[17] Easley, D., Kleinberg, J.: Networks, crowds, and markets: Reasoning about a highly connected world. Cambridge University Press, (2010).
[18] Gavin, A.C., Bosche M., Krause R., et al.: Functional organization of the yeast proteome by systematic analysis of protein complexes. Nature, vol. 415, no. 6868, pp. 141-147 (2002).
[19] Krogan, N.J., Cagney, G., Yu, H., et al.: Global landscape of protein complexes in the yeast Saccharomyces cerevisiae. Nature, vol. 440, no. 7084, pp. 637–643 (2006).
[20] Dezso, Z., Oltvai, Z.D., Barabasi, A. L.: Bioinformatics Analysis of Experimentally Determined Protein Complexes in the Yeast Saccharomyces cerevisiae. Genome Res, vol. 13, pp. 2450–2454 (2003).
[21] Brohee, S., and Helden, J.V.: Evaluation of clustering algorithms for protein-protein interaction networks. BMC bioinformatics, vol. 7, no. 1, pp. 1 (2006).
[22] Mewes, H.W., Frishman, D., et al.: MIPS: A database for genomes and protein sequences. Nucleic Acids Res, vol. 30, pp. 31-34 (2002).
[23] Chen, W.H., et al.: OGEE: an online gene essentiality database. Nucleic acids research, vol. 40, no. D1, pp. D901-D906 (2012).
[24] Jeong, H., Mason, S.P., Barabási, A.L.: Lethality and centrality in protein networks. Nature, vol. 411, no. 6833, pp. 41-2 (2001).
[25] Sengupta, S. M., et al.: The interactions of yeast SWI/SNF and RSC with the nucleosome before and after chromatin remodeling. Journal of Biological Chemistry, vol. 276, no. 16, 12636-12644 (2001).
[26] Tang, L., Nogales, E. and Ciferri, C.: Structure and function of SWI/SNF chromatin remodeling complexes and mechanistic implications for transcription. Progress in biophysics and molecular biology, vol. 102, no. 2, pp. 122-128 (2010).
[27] Imamura, Y, et al.: RSC Chromatin-Remodeling Complex Is Important for Mitochondrial Function in Saccharomyces cerevisiae. PloS one, vol. 10, no. 6, pp. e0130397, (2015).