Detection of Complexes in Biological Networks Through Diversified Dense Subgraph Mining

XIULI MA,¹ GUANGYU ZHOU,² JINGBO SHANG,³ JINGJING WANG,³
JIAN PENG,³ and JIAWEI HAN³

ABSTRACT

Protein–protein interaction (PPI) networks, providing a comprehensive landscape of protein interaction patterns, enable us to explore biological processes and cellular components at multiple resolutions. For a biological process, a number of proteins need to work together to perform a job. Proteins densely interact with each other, forming large molecular machines or cellular building blocks. Identification of such densely interconnected clusters or protein complexes from PPI networks enables us to obtain a better understanding of the hierarchy and organization of biological processes and cellular components. However, most existing graph clustering algorithms on PPI networks often cannot effectively detect densely connected subgraphs and overlapped subgraphs. In this article, we formulate the problem of complex detection as diversified dense subgraph mining and introduce a novel approximation algorithm to efficiently enumerate putative protein complexes from biological networks. The key insight of our algorithm is that instead of enumerating all dense subgraphs, we only need to find a small diverse subset of subgraphs that cover as many proteins as possible. The problem is modeled as finding a diverse set of maximal dense subgraphs where we develop highly effective pruning techniques to guarantee efficiency. To scale up to large networks, we devise a divide-and-conquer approach to speed up the algorithm in a distributed manner. By comparing with existing clustering and dense subgraph-based algorithms on several yeast and human PPI networks, we demonstrate that our method can detect more putative protein complexes and achieves better prediction accuracy.*

Keywords: complexes detection, dense subgraphs detection, diversification, graph clustering.

1. INTRODUCTION

RECENT DEVELOPMENTS in high-throughput experimental procedures have resulted in very large repositories of protein–protein interaction (PPI) and genetic interaction networks (Uetz et al., 2000; Ito et al., 2001; Gavin et al., 2006; Krogan et al., 2006). These interaction networks provide us a molecular landscape that defines fundamental biological processes in living cells. Many proteins need to interact closely

¹Key Laboratory of Machine Perception (MOE), School of EECS, Peking University, Beijing, China.

²Department of Computer Science, University of California, Los Angeles, California.

³Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois.

*This article was selected for oral presentation at RECOMB 2016 and an abstract is published in the conference proceedings.

with each other and form larger molecular machines to perform complex molecular functions. One major type of such molecular machines is the protein complex, in which proteins are densely packed together with very strong or even permanent interactions. As most cellular tasks are performed not by individual proteins or genes, but by groups of functionally associated proteins or genes (Adamcsek et al., 2006), identifying such tightly knit groups is crucial to understand and explore the structural and functional properties of biological networks. Thus, detection of protein complexes (Van Dongen, 2001; Bader and Hogue, 2003; Wu et al., 2009; Nepusz et al., 2012) from PPI networks is a fundamental problem in network biology, which has been normally formulated as a problem of finding subsets of densely interconnected proteins or clusters in networks.

A number of existing methods on complex detection pointed out protein complexes that correspond to densely connected regions or dense subgraphs in PPI networks (Bader and Hogue, 2003; King et al., 2004; Adamcsek et al., 2006; Liu et al., 2009; Nepusz et al., 2012). Most of them generally model the problem as graph clustering. They typically first partition a network into clusters and then postprocess or filter the clusters with density thresholds. Considering that proteins may have multiple functions and can belong to more than one complex, some of the algorithms have been proposed to detect overlapping clusters (Adamcsek et al., 2006; Ahn et al., 2010; Nepusz et al., 2012). However, no matter whether overlapping is considered, the objective function of such clustering methods is to maximize the difference between intra- and interconnectedness of clusters. Such models have no definitive level for what is sufficiently dense, which may lead to results that are not sufficiently dense, or lose some really dense and meaningful results.

Besides clustering-based methods, dense subgraph enumeration methods have also been explored to search complexes in PPI networks (Spirin and Mirny, 2003; Hu et al., 2005; Palla et al., 2005; Adamcsek et al., 2006; Liu et al., 2009). One line of research detects complexes based on the concept of clique, which is too restrictive that some potentially dense modules can be missed. We argue that this restriction should be relaxed, that is, a more flexible definition of density is required. Moreover, most methods of this kind aim to find all densely connected protein sets (Spirin and Mirny, 2003; Palla et al., 2005), while detecting the complete set of dense subgraphs is not practically tractable due to high computational complexity. In fact, it is often not necessary to output all dense subgraphs as substantial redundancy exists. The sheer number of all the dense subgraphs can be overwhelmingly large for a biologist to analyze. In fact, a majority of small subgraphs are completely covered by their supersets, which are also dense subgraphs. Therefore, dense subgraphs with maximal cardinality, which cannot be further extended, are preferred. However, another challenge we face is that even maximal dense subgraphs (MDSs) still can have large overlaps. While overlap is allowed between subgraphs, too much overlap introduces redundancy (Nepusz et al., 2012). Thus, we prefer a diverse set of dense subgraphs in the networks while the density is defined explicitly.

From the data mining point of view, most works on dense subgraph mining aim to find only one densest subgraph (Goldberg, 1984; Asahiro et al., 2000; Bahmani et al., 2012). They provide us with a global view of the dense part of a graph rather than the individual dense groups. Recently, mining algorithms can detect a small set of dense subgraphs (Tsourakakis et al., 2013). However, they cannot be directly adopted here as they only identify nonoverlapping dense subgraphs. Once a single densest subgraph is discovered, its nodes and edges will be removed from the network, and this process iterates until enough subgraphs are found or no edges are left. Another recent method (Balalau et al., 2015) can find subgraphs with maximum total density and limited overlapping. However, it defines density as the average degree of nodes instead of the density of edges in subgraphs, which may lead to large subgraphs with only modest density.

In this article, we propose to detect protein complexes by finding diversified dense subgraphs. With density measure, dense subgraphs that are protein complexes can be defined explicitly and detected flexibly at multiple resolutions. Instead of enumerating all the dense subgraphs, we output a small set of diversified MDSs. By *maximal*, we mean those complexes that are not subsets of any other dense subgraphs and thus cannot be further extended; by *diversified*, we mean a diverse set of dense subgraphs, which cover as many proteins as possible in the network. To the best of our knowledge, no existing work takes such diversification into the problem of complex detection. The key components of our algorithm are a set of efficient search trees that compactly traverse dense subgraphs. In particular, we present two strategies to construct or traverse the tree: depth-first search (DFS) and breadth-first search (BFS). In both strategies, we adopt a node-specific *potential* metric to guide the search process and develop efficient pruning techniques based on both density and diversity of the subgraphs. In this way, we embed the diversification of dense subgraphs into the enumeration of MDSs, which greatly improves the scalability of the algorithm. Finally, we further speed up the algorithm by parallelism to handle large-scale networks. We evaluate the effectiveness

and efficiency of our method extensively on several PPI networks from yeast to humans. Compared with the current state-of-the-art algorithms, our approach can get higher performance in most cases.

2. PROBLEM DEFINITION

Let $G=(V, E)$ be an undirected graph with a vertex set V and an edge set $E \subseteq V \times V$. $w(e)$ is the weight of edge $e \in E$. Weights are normalized to the range $[0, 1]$. We treat unweighted graphs as the special case where all weights are equal to 1. For a set of vertices $S \subseteq V$, we denote the subgraph of G induced by S as $G(S)=(S, E(S))$, where $E(S)=\{(u, v) \in E | u, v \in S\}$.

Definition 1. Density: The density of subgraph S , $den(S)$, is the ratio of the total weight of edges in $E(S)$ to the number of possible edges among $|S|$ vertices. The density of S is defined as $den(S)=\sum_{u,v \in S} w(u, v) / \binom{|S|}{2}$.

Note that if the graph is unweighted, the numerator is simply the number of actual edges. No matter whether weighted or not, the maximum possible density is 1.

Definition 2. Dense subgraphs: Given a graph G and a density threshold θ , $0 \leq \theta \leq 1$, an induced subgraph S of G is called a dense subgraph if it is connected and its density is no less than θ . A dense subgraph S is called a maximal dense subgraph (MDS) if there exists no dense subgraph S' in G such that $S \subseteq S'$.

In this article, we diversify MDSs both for efficiency and conciseness. Specially, we diversify the results to cover most nodes.

Definition 3. Coverage: Given a set of dense subgraphs $D=\{S_1, S_2, \dots\}$ in graph G , the coverage of D , denoted by $cov(D)$, is the number of nodes in G covered by the dense subgraphs in D , that is, $cov(D)=|\cup_{S_i \in D} S_i|$.

Problem Statement. Given a graph G , a density threshold θ , and an integer k , the problem of detecting k diversified MDSs is to discover a set of subgraphs, D , where $cov(D)$ is maximized, subject to each $S \in D$ is an MDS with density no less than θ in G and $|D| \leq k$. D is called the set of diversified MDSs.

Actually, by diversifying, our goal is to select a manageable subset of MDSs that are most representative according to the objective function coverage.

Figure 1 illustrates three complexes detected by our algorithm from the Krogan core dataset with density threshold 0.7. Shaded areas represent the dense subgraphs. Red, blue, and green nodes represent diverse nodes of the complexes, respectively; yellow nodes represent overlapped nodes. The set of two diversified MDSs is $\{A, C\}$.

Considering that it may be hard for users to specify a suitable value for k , we have an alternative problem definition.

Definition 4. Diversity: The diversity of a set of dense subgraphs D , $div(D)$, is the ratio of the coverage of D to the number of dense subgraphs in D : $div(D)=cov(D)/|D|$.

cov/k alternative problem statement. Given a graph G , a density threshold θ , and a diversity threshold γ , the problem cov/k is to discover a set D of dense subgraphs, where $|D|$ is maximum subject to $div(D) \geq \gamma$, and each $S \in D$ is an MDS with density no less than θ in G .

Note that for each different value of $|D|$, the goal of the problem cov/k is actually to maximize the coverage of D . So, the two problem statements have the same goal. Essentially, our focus is on maximizing the coverage subject to a cardinality constraint. Unfortunately, this problem is NP-hard (NP: nondeterministic polynomial time) (Feige, 1998). Note that a naive solution to the original problem is to first enumerate all the dense subgraphs, pick out the maximal ones, and then find k of them that cover most nodes in the graph by using the approximate greedy *Maxcover* algorithm (Nemhauser et al., 1978). However, such a solution is impractical when the graph is large because the number of maximal cliques is exponential to the graph size (Tomita et al., 2006; Eppstein et al., 2010). While maximal clique is a special case of MDS when it is fully connected, or its density is 1, the number of MDSs is even larger than that of maximal cliques. In this article, we integrate the searching and diversifying tightly into one unified process.

FIG. 1. Illustration of three complexes detected by our algorithm from the Krogan core dataset with density threshold 0.7. Shaded areas represent the dense subgraphs. Red, blue, and green nodes represent diverse nodes of the complexes, respectively; yellow nodes represent overlapped nodes. The set of two diversified maximal dense subgraphs (MDSs) is $\{A, C\}$.

3. METHODS

As enumerating all the dense subgraphs first is impractical, we take effort to diversify during enumerating, targeting at the dense subgraphs with both maximal cardinality and distinctive contribution on coverage. Before diving into the algorithm, let us first give a brief overview of the whole method.

3.1. Overview

As a whole, enumeration is a recursive backtracking procedure to recursively backtrack to add a vertex from the set of candidate vertices to grow the current subgraph S . A vertex v is a candidate to S if and only if v is a neighbor of any vertex in S . Specially, we adopt a node-specific *potential* metric to guide the search process and develop efficient pruning techniques based on properties of density and coverage.

3.1.1. Property of density. Note that in our definition of dense subgraphs, we can get a rather general while effective for pruning property.

Theorem 1. *Let G be a graph. Given a density threshold θ , if G' is a dense subgraph, there is at least one subset of G' whose density is no less than θ .*

Proof. Let G' be a dense subgraph. If we can identify one such subset G'' of G' and its density is no less than θ , then we prove the theorem. Let v^* be a vertex in G' with the degree no greater than the average of the degrees of vertices in G' , then we can prove that the density of $G' \setminus \{v^*\}$ is no less than the density of G' (Uno, 2010). The detailed proof is in the Appendix.

Note that this theorem introduces an adjacency relationship on dense subgraphs. Since any dense subgraph G' has such a vertex v^* , we can remove the vertices of G' iteratively until G' is empty, passing through only dense subgraphs. That is also to say, each dense subgraph can be incrementally constructed, one vertex at a time. The most important point is that we need an appropriate ordering of the vertices to ensure the monotone decreasing of density along the growth to prune with density.

3.1.2. Property of coverage. By diversifying, our goal is to select a manageable subset of MDSs that are most representative according to the objective function coverage. Many natural notions of representativeness satisfy submodularity; an intuitive notion of diminishing returns also is coverage (Badanidiyuru et al., 2014).

First, coverage is naturally associated with marginal gain,

$$\Delta cov(S|D) = cov(D \cup \{S\}) - cov(D) \quad (1)$$

where D is a set of MDSs and S is an MDS, which qualifies the increase in coverage obtained when adding S to set D .

Theorem 2. *The coverage function is monotone and submodular. Coverage is monotone as for all S and D , it holds that $\Delta cov(S|D) \geq 0$. Coverage is submodular as for all $A \subseteq B \subseteq D$ and $S \in D \setminus B$, the following diminishing returns condition holds:*

$$\Delta cov(S|A) \geq \Delta cov(S|B) \quad (2)$$

Proof. In Appendix.

As stated above, the problem of maximizing the coverage is NP-hard. Although a simple greedy *Maxcover* algorithm, which each time greedily selects a subgraph that adds maximum marginal coverage to the current set, is highly effective (Nemhauser et al., 1978), it requires random access to the elements, which means getting all the dense subgraphs ready first. As we desire diversifying during enumeration, it is impractical to use the greedy algorithm directly. The challenge now is that once a dense subgraph is constructed, we must immediately decide whether it has sufficient marginal value.

As observed by Badanidiyuru et al. (2014), the key reason why the classical greedy algorithm works is that at every iteration, an element is identified that reduces the gap to the optimal solution by a significant amount. In our scenario, if D_i is the set of first i MDSs picked by the algorithm, then the marginal gain of the next MDS S_{i+1} should be at least $(OPT - cov(D_i))/(k - |D_i|)$, where OPT is the coverage for the optimal solution and k is the cardinality constraint of the problem. In this way, we can sieve out those MDSs with large marginal values during enumeration.

However, this will require that we know (a good optimization to) the value of the optimal solution OPT , which is hard to obtain before getting the solution. Actually, to get a very crude estimate on OPT , it is enough to know the maximum marginal gain of any single result, $m = \max_{S \in D} cov(\{S\})$. Then, from monotony and submodularity, we have that $m \leq OPT \leq km$. Once we get this crude upper bound km on OPT , we can refine it.

Note that this assumes that the value m is known at the very beginning of the algorithm. That is to say, if we first find the maximum dense subgraph, we can pick out the dense subgraphs with enough marginal gain on the fly and thus diversify the results during enumeration. In this study, a maximum dense subgraph of a graph is a dense subgraph having maximum size, while an MDS is a dense subgraph that is not a subset of any other dense subgraph. The crucial point is that we must find the maximum dense subgraph first.

3.2. Detecting diversified MDSs

As analyzed above, knowing the maximum dense subgraph first will help us decide online whether any resulting subgraph has sufficient marginal gain. In this section, we will present a recursive backtracking procedure that enumerates the MDSs starting from each uncovered node by DFS, with highly pruning techniques based on both density and diversity to guarantee efficiency. In addition, we will propose an ordering among the nodes to guide the search while ensuring that the maximum dense subgraph is built first. Essentially, the trace of DFS from one node is a search tree, while the maximum dense subgraph corresponds to the leftmost branch in the search tree.

3.2.1. Finding the maximum dense subgraph first. With this goal on our mind, we adopt the greedy randomized adaptive search procedure (Abello et al., 2002). The problem of finding the maximum clique is NP-hard. In this study, we aim to find nonfully connected cliques, an even harder problem. As

finding the maximum dense subgraph is computationally intractable, it has been proposed to find the MDS. When an MDS is iteratively constructed, the choice of next vertex to be added can be determined by ordering all candidates by *potential* (Abello et al., 2002).

To introduce the notion of *potential*, we first give several preliminaries (Abello et al., 2002). A vertex x is called a θ -vertex with respect to a dense subgraph S if $G(S \cup \{x\})$ is a dense subgraph given the density threshold θ . The set of θ -vertices with respect to S is denoted by $N_\theta(S)$.

The potential of a subgraph R is

$$\phi(R) = |E(R)| - \theta \binom{|R|}{2} \quad (3)$$

The potential of a set R with respect to a disjoint set S is

$$\phi_S(R) = \phi(S \cup R) \quad (4)$$

Assume S is a dense subgraph. We seek a vertex $x \in N_\theta(S)$ to be added to S . One strategy for selecting x is to measure the effect of its selection on the *potential* of the other vertices in $N_\theta(S)$. To accomplish this, define the potential difference of a vertex $y \in N_\theta(S) \setminus \{x\}$ by selecting x to be

$$\delta_{S,x}(y) = \phi_{S \cup \{x\}}(\{y\}) - \phi_S(\{y\}) \quad (5)$$

The total effect, caused by the selection of x , on the *potential* of the remaining vertices in $N_\theta(S)$ is

$$\Delta_{S,x} = \sum_{y \in N_\theta(S) \setminus \{x\}} \delta_{S,x}(y) = |N_\theta(\{x\})| + |N_\theta(S)|(\deg(x)|_S - \theta(|S| + 1)) \quad (6)$$

We call this metric *potential* of x with respect to S for short. Actually, its exact meaning is the total potential difference cause by the selection of x on the remaining vertices of $N_\theta(S)$ (Abello et al., 2002). The vertex x that maximizes this metric is the one with a high number of θ -neighbors and with high degree with respect to S (where $\deg(x)|_S$ is the weighted degree of x with respect to S). A greedy algorithm that recursively selects such a vertex will eventually terminate with an MDS. As finding the maximum dense subgraph is intractable, we will use this MDS as the approximate maximum dense subgraph.

3.2.2. Enumerating with pruning. Enumerating all the MDSs starting from one node is a recursive backtracking procedure within which pruning is embedded. The *potential* metric of each node is used to guide the search process. Specially, at each internal node, we will compute the *potential* for each candidate of current subgraph and extend the candidates by the *potential* ordering. As stated above, the trace of DFS from one seed node is a search tree, while the approximate maximum dense subgraph corresponds to the leftmost branch in the search tree, which is obtained first. We will show that this order can facilitate the pruning by both density and diversity.

3.2.2.1. Pruning by density

Note that after we sort the candidates by the order of *potential*, a vertex can only appear in the subtree of another vertex with higher ordering, not vice versa. From Property 1, if the enumeration tree grows in this way, the density of the extending subgraph will decrease along the path in DFS; we can safely prune an expansion if the density is less than the density threshold.

Next, we discuss how to employ coverage to prune the search.

3.2.2.1. Pruning by diversifying

Note that after we sort the candidates by the order of *potential*, the leftmost branch of the search tree is the maximum dense subgraph, which is obtained first in the DFS. Once we get m , according to submodularity, we can get the crude bounds $m \leq OPT \leq km$ of OPT , so we can immediately refine it. Consider the following set:

$$O = \{(1 + \epsilon)^i | i \in \mathbb{Z}, m \leq (1 + \epsilon)^i \leq km\} \quad (7)$$

At least one of the thresholds $opt \in O$ should be a pretty good estimate of OPT , that is, there should exist at least some $opt \in O$ such that $(1 - \epsilon)OPT \leq opt \leq OPT$. Based on the analysis of Wang et al. (2013), on the one hand, we will get its upper bound. On the other hand, we want to have the threshold lower bounded.

Assume we have found a dense subgraph S and want to decide whether we should expand it further. Let d be the depth of its subtree. Remember that we have got an MDS at first, here we use it as the upper bound of d , $\bar{d}$, which is the size of the MDS. Then, the size of each dense subgraph to be generated from S is at most $\bar{d}$. Assume S has already got l distinct vertices. As S has at most $(\bar{d} - |S|)$ nodes to expand, its marginal gain is at most $l + (\bar{d} - |S|)$.

On the other hand, we want to have its threshold lower bounded. Based on the analysis of Badanidiyuru et al. (2014) and the analysis at the beginning of this section, the marginal gain of each result should be at least $(opt - cov(D))/(k - |D|)$, where D is the set of MDSs we have got, opt is an estimate for OPT , which is the coverage for the optimal solution, and k is the cardinality constraint of the problem. The point is, as we have to decide immediately whether to prune current expansion or not and cannot try different or multiple values of opt , we must fix a value for opt . Based on extensive experiments, it can be shown, with a value between 10% and 30% of km , to be opt , the best effectiveness scores can be obtained. However, if opt is set to be the maximal value, which is km , the efficiency improves a lot, while effectiveness remains rather similar with only limited decreasing. With this value, we can determine the threshold for marginal gain. We also find that the maximal value of opt , which is km , leads to the highest threshold, thus the strictest pruning, which has the same performance scores as retaining only the leftmost branch while pruning all the other branches in each search tree. For higher efficiency, we use the maximal value opt , km , in most evaluations. [Note that in some cases, as analyzed by Badanidiyuru et al. (2014), opt is set to a lower threshold as βopt with $\beta = 1/2$.]

In summary, in an internal vertex, we will prune a subtree if its predicted biggest marginal gain is lower than the threshold. It should be mentioned that for further diversifying, we also execute *Maxcover* later on all the unpruned MDSs as postprocessing.

A step-by-step description of the growth process starting from S is as follows. Step 1: Find all the candidates. Step 2: Order all the candidates according to *potential*. Step 3: For each candidate r (in the decreasing order), $S' = S \cup \{r\}$. If $den(S') \geq \theta$ and the predicted marginal gain is no less than the threshold, let $S_{t+1} = S'$ and iteratively grow from S_{t+1} . Otherwise, declare S' an MDS. The pseudocode of the algorithm is illustrated in Algorithm 2. We call this method MDS-DFS.

3.3. Breadth-first search

As the DFS method sorts the candidates based on *potential* at every step, which may hurt the efficiency, we explore the BFS method to further speed up the method. A step-by-step description of the growth process starting from S is as follows. Step 1: Find all the candidates of the current subgraph. Step 2: Order all the candidates according to *potential* order. Step 3: Expand the candidates directly in the decreasing order, until the density is less than the threshold, and then get all the candidates and return to Step 2. Declare an MDS until there are no more candidates. The difference with MDS-DFS lies in expanding without iteration at each step, and only one MDS is obtained from one source node. We call this method MDS-BFS.

3.4. Postprocessing by Maxcover

From different seeds, we may get some overlapping MDSs. So, after finding all the MDSs, we can postprocess them by *Maxcover* (Nemhauser et al., 1978) for further diversifying. It starts with the empty set $D_0 = \emptyset$, and at each iteration i over the whole set of MDSs, it chooses an MDS S maximizing (1), that is, $D_i = D_{i-1} \cup \arg\max_S cov(S|D_{i-1})$. Nemhauser et al. (1978) prove that this greedy algorithm can obtain a $(1 - 1/e) = 0.63$ approximation. In summary, pruning by both density and diversity is performed when the process is starting a new search subtree, whereas *Maxcover* procedure is executed on all the nonpruned branches of all the search trees. This is postprocessing for further diversifying.

When k is hard to prespecify, we propose the cov/k solution for diversifying. Essentially, cov/k is based on *Maxcover*. Starting with k being 1, in each iteration, once an MDS is chosen to maximize the coverage, the value of cov/k is compared with the diversity threshold. Stop when the threshold is broken. We call the MDS method with postprocessing cov/k as MDS+ $covk$.

3.5. Outline of the whole algorithm

Our algorithm grows dense subgraphs from seeds. Initially, it sorts the nodes as seeds by the global degrees or the sum of weighted connections in weighted graph. The algorithm grows the dense subgraphs

from the first seed based on DFS or BFS. Whenever the growth process finishes, the algorithm selects the next seed by considering all the nodes that have not been covered in any of the dense subgraphs found so far. The entire procedure terminates when there are no nodes remaining to consider.

The main procedure is as illustrated in Algorithm 1. In Algorithm 2, we illustrate the enumeration based on DFS, that is, MDS-DFS.

The main procedure MDS is as follows:

Algorithm 1: MDS

```

input:  $G(V, E)$ ,  $\theta$  (density threshold),  $k$ 
output:  $D$ , List of maximal dense subgraphs
SortByDeg( $V$ )
foreach  $v_i \in V$  not covered by any MDS in  $D$  do
     $firstBranch = TRUE$ 
     $enum(\{v_i\}, Neighbor(\{v_i\}), G(V, E), \theta, k, D)$ 
end
return  $D$ 

```

The subprocedure Enumeration is as follows:

Algorithm 2: Enumeration

```

 $enum(S, R, G(V, E), \theta, k, D)$ 
input:  $S$  (current subgraph),  $R$  (remaining candidate sets),  $G(V, E)$ ,  $\theta$ ,  $k$ ,  $D$ 
 $isLeaf = True$ 
if  $R.size \neq 0$  then
    graspSort( $S, R$ )
    foreach  $r \in R$  do
         $S' = S + \{r\}$ 
         $R' = Neighbor(S') - S'$ 
        if  $density(S') > \theta$  then
            if not  $firstBranch$  then
                if  $shouldPruning(S', m, k, D)$  continue
            end
             $isLeaf = False$ 
             $enum(S', R', G(V, E), \theta, k, D)$ 
        end
    end
end
if  $isLeaf$  then
    if  $firstBranch$  then
         $m = S.size$ 
         $firstBranch = False$ 
    end
     $D = D + \{S\}$ 
end

```

Figure 2 illustrates the MDS detection process. The density threshold is set to 0.8. k is 4, which means four diversified MDSs are desired. From the seed E , the algorithm greedily generates the maximum dense subgraph EDBG. Thus, m is 4. opt is set to be $km/2$. After that the MDSs, EDBC, EDGH, and EGHA, are generated. All the other branches are pruned because they do not bring diverse contribution.

3.6. Computational complexity of the MDS algorithm

If the network is extremely dense (e.g., fully connected with weight 1.0) and the density threshold is extremely low (e.g., 0.0), the DFS will exhaust all possible subgraphs. Therefore, the worst-case theoretical time complexity is exponential. However, real PPI networks are quite sparse. As a result, in practice, our algorithm is very efficient. MDS-BFS is more efficient than MDS-DFS. The complexity of

FIG. 2. Illustration of the MDS detection process.

BFS search of MDS-BFS is $O(n)$, n is the number of nodes in the graph. Moreover, when the graph is tremendously large, we also provide the partition-enabled MDS, which is based on graph partitions, thus allowing full parallelism.

The problem of finding the maximum clique is NP-hard. In this study, we aim to find nonfully connected dense subgraphs, an even harder problem. Although our algorithms cannot be guaranteed to find all the solutions, they are efficient when applied to relatively sparse graphs such as a network of protein interactions.

4. SCALING UP THE ALGORITHM BASED ON PARTITIONING

For a large-scale graph, we propose a divide-and-conquer approach to speed up the algorithm in a distributed manner. We will first partition the graph into several pieces. On each piece, the mining algorithm is executed to discover diversified MDSs. After that the results from each partition are merged into one set to be diversified. For the partitioning, we use the measure *cohesiveness* (Nepusz et al., 2012) to roughly cut the graph into several parts, each of which being no bigger than a size limit. When merging the results from each partition, we first extend each MDS based on density to avoid the effect of partitioning. If one subgraph has been covered by 80% of another extended subgraph, we will not retain it in the result set. After that we merge the extended results from each partition into one set and diversify it by cov/k . The detection on each partition can be in parallel. The main procedure of scaling up MDSs based on partitioning is in the Appendix.

5. RESULT

In this section, we evaluate the effectiveness and efficiency of our method, compared with state-of-the-art methods on a range of biological networks from different species.

5.1. Experiment setup

5.1.1. Datasets. We use several real-world networks from two different organisms, one category is five PPI datasets for yeast and the other is the genetic network for humans. The statistics of these datasets are in Table 1.

5.1.2. Gold standards. For yeast PPI networks, we compare the detected complexes with four reference complex sets: the first derived from the Munich Information Center for Protein Sequence (MIPS) catalog of protein complexes and the second from gene ontology-based complex annotations in the

TABLE 1. STATISTICS OF PROTEIN-PROTEIN INTERACTION DATASETS: THE NUMBER OF VERTICES AND EDGES

	<i>Gavin</i>	<i>Krogan core</i>	<i>Krogan extended</i>	<i>Collins</i>	<i>BioGRID</i>	<i>String</i>
$ V $	1855	2708	3672	1622	5640	16,384
$ E $	7669	7123	14,317	9074	59,748	419,383
Weighted	Y	Y	Y	Y	N	Y

Saccharomyces Genome Database (SGD). The third one COMPLEAT is from www.flymai.org/compleat. We also merge these three sets into one whole set named Merged. For the Human String network, the ground truth String-GT is gained from: <http://mips.helmholtz-muenchen.de/genre/proj/corun>. Table 2 illustrates some properties of the gold standards.

5.1.3. Evaluation metrics. We assess the quality of the predicted complexes by three scores defined by Nepusz et al. (2012): fraction (*frac*), accuracy (*acc*), and the maximum matching ratio (*mmr*). *frac* is defined to be the fraction of pairs between predicted and reference complexes with an overlap score no less than 0.25; *acc* is the geometric mean of two other measures, namely the clustering-wise sensitivity (*Sn*) and the clustering-wise positive predictive value (PPV). See Nepusz et al. (2012) for a precise description. *mmr* is based on a maximal one-to-one mapping between predicted and reference complexes. With a bipartite graph, in which the two sets of nodes represent the reference and predicted complexes, respectively, and an edge is weighted by the overlap score between two complexes, *mmr* is the total weight of the selected edges, which represent an optimal one-to-one matching between the two sets, divided by the number of reference complexes. Besides these metrics, we also do comparisons on number of the resulted MDSs and coverage.

5.1.4. Baseline for comparison. We choose the most prominent methods for detecting protein complexes, MCODE (Bader and Hogue, 2003), RNSC (King et al., 2004), MCL (Van Dongen, 2001; Van Dongen, 2008), AP (Frey and Dueck, 2007), CMC (Liu et al., 2009), COACH (Wu et al., 2009), and ClusterONE (Nepusz et al., 2012) as the baselines. For easy understanding, we first briefly review ClusterONE. For the other baselines, please refer to the Other Baseline Methods section in the Appendix for detailed introduction as by Nepusz et al. (2012).

ClusterONE consists of three major steps (Nepusz et al., 2012). First, starting from a single seed vertex, a greedy procedure adds or removes vertices to find groups with high cohesiveness. The growth process is repeated from different seeds to form multiple possibly overlapping groups. Second, merge those pairs of groups for which the overlap score is above a specified threshold. Third, discard complex candidates that contain less than three proteins or whose density is below a given threshold. Note that the density metric is used as a postprocessing filter of the method, which is not as direct as ours.

5.2. Evaluations

5.2.1. Effectiveness. In this section, we present the number of results and coverage (the number of nodes covered) for AP, MCL, ClusterONE, and our method on different datasets under different density thresholds. The evaluation is under density thresholds between 0.5 and 0.9. We visualize density 0.6 for number of results in Figure 3a and coverage in Figure 3b. The detailed results for other thresholds are shown in Supplementary Tables S1 and S2 in Appendix. As shown, our method finds more complexes and covers more nodes than all the other methods on almost all the datasets and density thresholds. This is because that our method is based on enumeration with smart ordering based on candidates, which can discover more results. However, pruning can still guarantee efficiency, and diversifying the results can remove redundancy as well. For all the datasets except BioGRID, MDS represents using our method MDS-

TABLE 2. STATISTICS OF THE GOLD STANDARDS

	<i>MIPS</i>	<i>SGD</i>	<i>COMPLEAT</i>	<i>Merged</i>	<i>String-GT</i>
Number of nodes	1189	1279	2877	2751	1700
Number of complexes	203	323	1317	957	840
Overlapping complex pairs	401 (2%)	296 (0.6%)	9845 (1.1%)	8468 (1.8%)	7443 (2%)

FIG. 3. Number of results (a) and coverage (b) for different methods on all the datasets under density threshold $\theta = 0.6$.

DFS with leftmost branch only (the strictest pruning) and MDS+covk represents $cov/k \geq 4$. Comparing MDS with MDS+covk, we can find that on such datasets as Krogan core, Krogan extended, and Collins, MDS+covk can cover almost the same number of nodes with fewer results. On String, MDS+covk can cover almost 89% nodes with 33% fewer results. That means, by diversifying, a concise set of results can capture the most essential information. For unweighted network BioGRID, a different and higher density threshold is set, as also suggested by Nepusz et al. (2012). For BioGRID, the parameters are set such that each method can get its highest performance as shown in Supplementary Tables S1 and S2 in the Appendix.

We also calculate the general statistics for overlapping MDS pairs and overlap size distribution of our method as illustrated in Table 3. All the five datasets are binarized unweighted datasets, as in Supplementary Table S3 in the Appendix. The density threshold is set to be 0.6. The fraction of overlapping MDS pairs is close to that observed in SGD and COMPLEAT gold standard. For BioGRID, MDSs have more overlaps in total, but fewer big overlaps (of size being two or higher) than ClusterONE (Nepusz et al., 2012).

5.2.2. Performance. We present the three quality scores on the weighted datasets based on MIPS, SGD, COMPLEAT, and the Merged reference set in Figure 4. Shades of the same color denote individual quality scores; the total height of each bar is the value of the composite score, the sum of the three scores (Nepusz et al., 2012). Larger scores are better. For all the metrics, we compare under different density thresholds from 0.5 to 0.9 with AP, MCL, and ClusterONE. For AP and MCL, which cannot use density as a parameter, density is used as a postprocessing filter as in ClusterONE. We visualize the results for all the ground truths under 0.6 in Figure 4. The results for other thresholds are quite similar. We also visualize the results under 0.7 in Figure 5. It can be observed that the higher the density threshold, the better our algorithm performance than the baselines. The density threshold is typically set as 0.7 (Wu et al., 2009). We also illustrate detailed individual scores for all the methods with their best parameter settings in Supplementary Table S4 in the Appendix. The results on String are illustrated in Supplementary Table S5. As shown, we are capable of matching more complexes and providing better one-to-one mapping with reference complexes in almost all the datasets except Collins compared with the baselines.

TABLE 3. GENERAL STATISTICS AND OVERLAP SIZE DISTRIBUTION BETWEEN OVERLAPPING MAXIMAL DENSE SUBGRAPH PAIRS

Dataset	Gavin		Krogan core		Krogan extended		Collins		BioGRID	
Number of MDSs	502		662		954		275		925	
Overlapping pairs	1.10%		0.54%		0.68%		1.38%		16.53%	
Overlap size	#pairs	Percentage	#pairs	Percentage	#pairs	Percentage	#pairs	Percentage	#pairs	Percentage
1	636	45.95	629	53.49	1781	57.86	153	29.42	61,198	86.53
2	306	22.11	298	25.34	619	20.11	116	22.31	6277	8.88
3	140	10.12	78	6.63	205	6.66	0	11.54	1561	2.21
4	100	7.23	46	3.91	131	4.26	31	5.96	758	1.07
5 or larger	202	14.60	125	10.63	342	11.11	160	30.77	927	1.31

MDS, maximal dense subgraph.

FIG. 4. Comparison of the performance of various methods on four PPI weighted datasets using different gold standards under density threshold $\theta=0.6$. Colors correspond to the various algorithms, various shades of the same color denote the individual components of the composite score of the algorithm (*dark*=*frac*, *medium*=*acc*, *light*=*mmr*). The total height of each column is the value of the composite score for a given algorithm on a given dataset. PPI, protein–protein interaction. (a) MIPS gold standard; (b) SGD gold standard; (c) complete; (d) merged groundTruth.

We also compare the scores on unweighted network, which is the binary version of the weighted datasets, illustrated in Figure 6a, b. For all the baselines, we use their best parameter settings. Supplementary Table S3 in the Appendix contains the detailed individual scores for Figure 6b. As illustrated, MDS gets the highest performance scores on most datasets except Collins. For BioGrid, we only show the results of those baselines, which can finish it efficiently. We also visualize the results under 0.7 in Figure 7. It can be observed that the performance of the other baselines decreases very rapidly and dramatically with increasing density threshold and eventually cannot even detect the complexes.

For the PPV score, as mentioned in supplementary material of Nepusz et al. (2012), a perfect clustering algorithm that always returns the reference complexes from the data may have a lower PPV than a dummy algorithm, which places every protein in a separate cluster.

FIG. 5. Comparison of the performance (bottom-up: frac, acc, mmr) of various methods on four PPI weighted datasets using different gold standards under density threshold $\theta = 0.7$. (a) MIPS gold standard; (b) SGD gold standard; (c) complete; (d) merged groundTruth.

The accuracy measure explicitly penalizes predicted complexes that do not match any of the reference complexes. However, gold standard sets of protein complexes are often incomplete (Nepusz et al., 2012). As a consequence, predicted complex not matching any known reference complexes may still exhibit high functional similarity or be highly colocalized and therefore they could still be prospective candidates for further in-depth analysis. In other words, a predicted complex that does not match a reference complex is not necessarily an undesired result. We evaluate the functional homogeneity of some detected complexes by literature search as follows.

We find the following complex in Gavin under density threshold 0.3: (YBR265W YPR035W YDR028C YBR126C YNL076W YMR261C YBL023C YML100W YDR074W), which cannot be found in any gold standard.

We search the literature and find this complex included in the excel file 1752-0509-3-74-S3.XLS from this link: “Set of 491 complexes originating from Gavin et al., Nature 2006.” It is included in trehalose-6-

FIG. 6. Comparison of the performance (bottom-up: frac, acc, mmr) of various methods on five PPI unweighted datasets (the binarized version of Collins, Krogan core, Krogan extended and Gavin, and BioGRID) using (a) COMPLEAT and (b) Merged gold standard. The parameters are set individually such that each method can get its highest composite score of performance.

phosphate synthase/phosphatase complex: YBL023C YBR265W YNL076W YPR035W YBR126C YDR028C YDR074W YDR171W YJL138C YJR138W YKR059W YLR075W YML100W YMR261Cs

It also appears in the “Process GO Term physiological processes and related genes” from this link. It gives out the set of interacting genes of YBR126C, which includes all the proteins in the above complex we found.

5.2.3. Performance on String genetic network. We present the results on String dataset. By partitioning algorithm, we first partition the dataset and discover the MDSs in each partition, and then merge the results into one whole and then do diversifying (cov/k). We present our performance score of String dataset with different baseline methods in Figure 8.

I is the parameter inflation for MCL algorithm, which tunes the granularity of clustering. Larger inflation values result in smaller clusters, while smaller inflation values generate only a few large clusters. For MCL,

FIG. 7. Comparison of the performance (bottom-up: frac, acc, mmr) of various methods on four PPI unweighted datasets under density threshold $\theta = 0.7$ using (a) COMPLEAT and (b) Merged gold standard.

FIG. 8. Comparison of the performance of MCL, ClusterONE, MDS, and MDS+covk on String datasets using the String-GT gold standard.

I is set to be 3.5. For ClusterONE, the density threshold is set to be 0.6. For MDS, the density threshold is set to be 0.8. For MDS+covk, cov/k is set to be 4.

5.2.4. Pruning. We evaluate the effectiveness and efficiency of MDSs with or without pruning. In Figure 9, we compare the performance score and the running time of our algorithm with different diversity thresholds by setting opt to be different percentages of km . In this evaluation, the dataset is Gavin, the density threshold is 0.6, and k is 300. When opt is set to 0, enumeration is done without any pruning by diversity. The results are all the MDSs. When opt is set to 100% of km , enumeration is actually done with the strictest pruning. As we can see, with pruning, efficiency can be improved a lot, while effectiveness remains rather same. So, in most experiments, we run enumeration with the strictest pruning; only the leftmost branches were traced. That means, at each step, we only select the candidate with the highest *potential* value to extend.

5.2.5. Scalability. We evaluate the scalability of our algorithm on various datasets. On small- to medium-sized networks, both our method and other baselines can finish in seconds, so we do not illustrate them. However, on a very big network, such as String, some baselines can not finish in time. We execute the partitioning-based MDS in that scenario and can finish in time.

For the partitioning-based MDS, the number of partitions in our evaluation is illustrated in Table 4. The size and number can be easily adjusted for different computing systems.

5.2.6. Summary. From the above experimental evaluations, we can see that our algorithm can detect and explore the complexes flexibly at different densities. In almost all the PPI networks, our algorithms are capable of matching more complexes with a high accuracy and providing good one-to-one mapping with reference complexes. On very dense datasets, such as Collins, our algorithm may get the same or a little lower performance than some baselines. However, if the density threshold is set to 0.7 or higher, our algorithm outperforms all the evaluated methods. As, in general, known complexes tend to have high density in the PPI networks, it is reasonable that the density threshold is typically set as 0.7 (King et al., 2004; Altaf-UI-Amin et al., 2006; Wu et al., 2009). It can also be observed that the performance of the other

FIG. 9. Comparison of pruning with different diversity thresholds by setting different value of opt .

TABLE 4. PARTITION NUMBERS FOR DIFFERENT DATASETS

	<i>Gavin</i>	<i>Krogan core</i>	<i>Krogan extended</i>	<i>Collins</i>	<i>BioGRID</i>	<i>String</i>
No. Partitions	220	639	1241	254	1483	4725

baselines decreases very rapidly and dramatically with increasing density threshold and eventually cannot even detect the dense complexes hidden in the PPI networks. For the diversity threshold, cov/k , three to five is an appropriate option.

6. DISCUSSION

We have modeled the problem of detecting complexes in biological networks as discovering the diversified MDSs. With the edge density measure, the dense subgraphs, which are complexes, can be defined explicitly and flexibly. Based on this, we seek all the dense subgraphs of maximal cardinality with at least the specified density. Meanwhile, results without enough diversity are sieved along the enumeration. We also scale up the algorithm by the partitioning method. We evaluate the effectiveness and efficiency of our method on a diverse set of interaction networks from different species, five PPI networks for yeast and one genetic network for humans. Extensive experiments show that our results have better correspondence with reference complexes in MIPS, SGD, COMPLEAT, and the ground truth database for String than the state of the art.

In the future, we plan to pursue further improvements in larger-scale and denser networks. We believe the algorithm is general in various domains, such as identifying groups of densely interconnected nodes in social networks or word association networks. We also hope to explore other biological network-based applications, such as identifying homology relationships between sequences and orthology inference across multiple species.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China [61103025 to X.M.] and the China Scholarship Council [201308110276 to X.M.]. This work was sponsored, in part, by the National Science Foundation [IIS-1017362, IIS-1320617, and IIS-1354329 to J.H.] and NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov) [1U54GM114838 to J.H.].

AUTHOR DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

- Abello, J., Resende, M.G., and Sudarsky, S. 2002. Massive quasi-clique detection. In *LATIN 2002: Proceedings: 5th Latin American Symposium on Theoretical Informatics*, Cancun, Mexico, April 2002. Rajsbaum, S. (ed.), LNCS 2286. Springer-Verlag, Berlin/Heidelberg. Pgs. 598–612.
- Adamcsek, B., Palla, G., Farkas, I.J., et al. 2006. Cfinder: Locating cliques and overlapping modules in biological networks. *Bioinformatics* 22, 1021–1023.
- Ahn, Y.-Y., Bagrow, J.P., and Lehmann, S. 2010. Link communities reveal multiscale complexity in networks. *Nature* 466, 761–764.
- Altaf-UI-Amin, M., Shinbo, Y., Mihara, K., et al. 2006. Development and implementation of an algorithm for detection of protein complexes in large interaction networks. *BMC Bioinformatics* 7, 207.
- Asahiro, Y., Iwama, K., Tamaki, H., et al. 2000. Greedily finding a dense subgraph. *J. Algorithms* 34, 203–221.
- Badanidiyuru, A., Mirzasoleiman, B., Karbasi, A., et al. 2014. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA. Pgs. 671–680.

- Bader, G.D., and Hogue, C.W. 2003. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics* 4, 2.
- Bahmani, B., Kumar, R., and Vassilvitskii, S. 2012. Densest subgraph in streaming and mapreduce. *Proceedings of the VLDB Endowment* 5, 454–465.
- Balalau, O.D., Bonchi, F., Chan, T., et al. 2015. Finding subgraphs with maximum total density and limited overlap. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining* pages. ACM, 379–388.
- Eppstein, D., Löffler, M., and Strash, D. 2010. Listing all maximal cliques in sparse graphs in near-optimal time. In: *International Symposium on Algorithms and Computation (ISAAC 2010). Lecture Notes in Computer Science*, vol. 6506. Cheong, O., Chwa, K.Y., and Park, K. (eds.). Springer, Berlin/Heidelberg.
- Feige, U. 1998. A threshold of $\ln n$ for approximating set cover. *JACM* 45, 634–652.
- Frey, B.J., and Dueck, D. 2007. Clustering by passing messages between data points. *Science* 315, 972–976.
- Gavin, A.-C., Aloy, P., Grandi, P., et al. 2006. Proteome survey reveals modularity of the yeast cell machinery. *Nature* 440, 631–636.
- Goldberg, A.V. 1984. *Finding a maximum density subgraph*. University of California, Berkeley, CA.
- Hu, H., Yan, X., Huang, Y., et al. 2005. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics* 21(Suppl.1), i213–i221.
- Ito, T., Chiba, T., Ozawa, R., et al. 2001. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl. Acad. Sci.* 98, 4569–4574.
- King, A.D., Pržulj, N., and Jurisica, I. 2004. Protein complex prediction via cost-based clustering. *Bioinformatics* 20, 3013–3020.
- Krogan, N.J., Cagney, G., Yu, H., et al. 2006. Global landscape of protein complexes in the yeast *saccharomyces cerevisiae*. *Nature* 440, 637–643.
- Liu, G., Wong, L., and Chua, H.N. 2009. Complex discovery from weighted ppi networks. *Bioinformatics* 25, 1891–1897.
- Nemhauser, G.L., Wolsey, L.A., and Fisher, M.L. 1978. An analysis of approximations for maximizing submodular set functions. *Math Prog* 14, 265–294.
- Nepusz, T., Yu, H., and Paccanaro, A. 2012. Detecting overlapping protein complexes in protein-protein interaction networks. *Nat Methods* 9, 471–472.
- Palla, G., Derényi, I., Farkas, I., et al. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 814–818.
- Spirin, V., and Mirny, L.A. 2003. Protein complexes and functional modules in molecular networks. *Proc. Natl. Acad. Sci. U. S. Am.* 100, 12123–12128.
- Tomita, E., Tanaka, A., and Takahashi, H. 2006. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theor. Comput. Sci.* 363, 28–42.
- Tsourakakis, C., Bonchi, F., Gionis, A., et al. 2013. Denser than the densest subgraph: Extracting optimal quasi-cliques with quality guarantees. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Dhillon, I., Koren, Y., and Ghari, R., et al. (eds.). ACM, New York, NY, USA. Pgs. 104–112.
- Uetz, P., Giot, L., Cagney, G., et al. 2000. A comprehensive analysis of protein–protein interactions in *saccharomyces cerevisiae*. *Nature* 403, 623–627.
- Uno, T. 2010. An efficient algorithm for solving pseudo clique enumeration problem. *Algorithmica* 56, 3–16.
- Van Dongen, S. 2008. Graph clustering via a discrete uncoupling process. *SIAM J. Matrix Anal. Appl.* 30, 121–141.
- Van Dongen, S.M. 2001. Graph clustering by flow simulation. PhD Dissertation, University of Utrecht.
- Wang, J., Cheng, J., and Fu, A.W.-C. 2013. Redundancy-aware maximal cliques. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* pages 122–130. ACM.
- Wu, M., Li, X., Kwoh, C.-K., et al. 2009. A core-attachment based method to detect protein complexes in ppi networks. *BMC Bioinformatics* 10, 169.

Address correspondence to:

Dr. Xiuli Ma

Key Laboratory of Machine Perception (MOE)

School of EECS

Peking University

Room 2311, Science Building No.2

No.5 Yiheyuan Road, Haidian District

Beijing 100871

China

E-mail: xlma@pku.edu.cn

Appendix

PROOF OF THEOREM 1

Theorem 3. Let G be a graph. Given a density threshold θ , if G' is a dense subgraph, there is at least one subset of G' whose density is no less than θ .

Proof. Let G' be a dense subgraph. $G' = (V', E')$. If we can identify one such subset G'' of G' and its density is no less than G' , then we prove the theorem. Let v^* be a vertex in G' with the degree no greater than the average of the degrees of the vertices in G' . E_1 is the set of edges in G' incident to v^* . E_2 is the set of edges $E_2 \subseteq V' \setminus \{v^*\} \times V' \setminus \{v^*\}$. Then, we have

$$\begin{aligned}
 & \text{den}(G' \setminus \{v^*\}) - \text{den}(G') \\
 &= \frac{\sum_{(u,v) \in E_2} w(u,v)}{\frac{(|V'| - 1)(|V'| - 2)}{2}} - \frac{\sum_{(u,v) \in E'} w(u,v)}{\frac{|V'| \cdot (|V'| - 1)}{2}} \\
 &= \frac{2}{|V'| - 1} \cdot \frac{|V'| \cdot \sum_{(u,v) \in E_2} w(u,v) - (|V'| - 2) \cdot \sum_{(u,v) \in E'} w(u,v)}{|V'| \cdot (|V'| - 2)} \\
 &= \frac{2}{|V'| - 1} \cdot \frac{|V'| \cdot \sum_{(u,v) \in E_2} w(u,v) - |V'| \cdot \sum_{(u,v) \in E'} w(u,v) + 2 \cdot \sum_{(u,v) \in E'} w(u,v)}{|V'| \cdot (|V'| - 2)} \\
 &= \frac{2}{|V'| - 1} \cdot \frac{2 \cdot \sum_{(u,v) \in E'} w(u,v) - |V'| \cdot \sum_{(u,v) \in E_1} w(u,v)}{|V'| \cdot (|V'| - 2)} \\
 &= \frac{2}{|V'| - 1} \cdot \frac{\frac{2 \cdot \sum_{(u,v) \in E'} w(u,v)}{|V'|} - \sum_{(u,v) \in E_1} w(u,v)}{|V'| - 2} \\
 &= \frac{2}{|V'| - 1} \cdot \frac{\text{AvgWeight}(G') - \text{weight}(v^*)}{|V'| - 2} \geq 0
 \end{aligned}$$

PROOF OF THEOREM 2

Theorem 4. The coverage function is monotone and submodular. Coverage is monotone as for all S and D , it holds that $\Delta \text{cov}(S|D) \geq 0$. Coverage is submodular as for all $A \subseteq B \subseteq D$ and $S \in D \setminus B$, the following diminishing returns condition holds:

$$\Delta \text{cov}(S|A) \geq \Delta \text{cov}(S|B) \quad (8)$$

Proof.

Monotone. Coverage is monotone if for all S and D , it holds that $\Delta \text{cov}(S|D) \geq 0$. As when a set D of MDSs has been selected, adding one more MDS can always cover no fewer nodes, that is, leading to non-negative marginal gain. So, $\Delta \text{cov}(S|D) \geq 0$. For every $A \subseteq B \subseteq D$, $\text{cov}(B) - \text{cov}(A) = \text{cov}(A \cup \{B \setminus A\}) - \text{cov}(A) \geq 0$. $\text{cov}(B) \geq \text{cov}(A)$. So, coverage is monotone.

Submodularity. Coverage is submodular if for every $A \subseteq B \subseteq D$ and $S \in D \setminus B$, it holds that $\Delta \text{cov}(S|A) \geq \Delta \text{cov}(S|B)$. As $A \subseteq B$, S always has no less overlap with B than A , $\text{cov}(A \cup \{S\}) - \text{cov}(A) \geq \text{cov}(B \cup S) - \text{cov}(B)$. So, $\Delta \text{cov}(S|A) \geq \Delta \text{cov}(S|B)$. This is the diminishing returns effect. Figure 10 illustrates this effect in the MDS diversification application. In our scenario, the marginal gain provided by selecting an MDS S' given that we have selected $\{S_1, S_2\}$ does not increase as we select MDS $\{S_3\}$.

FIG. 10. Illustration of the diminishing returns effect in context of diversifying MDSs. The blue regions indicate nodes covered by the selected MDSs. The red nodes in the red region indicate the additional coverage by adding a new MDS S' . If more MDSs have been already selected, there is more overlap, hence less gain in coverage: $\Delta cov(S'|\{S_1, S_2\}) \geq \Delta cov(S'|\{S_1, S_2, S_3\})$. Adding S' in (b) gets less gain than in (a).