Finding Dense and Connected Subgraphs in Dual Networks

Yubao Wu[1], Ruoming Jin[2], Xiaofeng Zhu[3], Xiang Zhang[1]
[1]*Department of Electrical Engineering and Computer Science, Case Western Reserve University,*
[2]*Department of Computer Science, Kent State University,*
[3]*Department of Epidemiology and Biostatistics, Case Western Reserve University,*
[1]{*yubao.wu, xiang.zhang*}*@case.edu,* [2]*jin@cs.kent.edu,* [3]*xxz10@case.edu*

***Abstract*—Finding dense subgraphs is an important problem that has recently attracted a lot of interests. Most of the existing work focuses on a single graph (or network[1]). In many real-life applications, however, there exist *dual* networks, in which one network represents the physical world and another network represents the conceptual world. In this paper, we investigate the problem of finding the densest connected subgraph (DCS) which has the largest density in the conceptual network and is also connected in the physical network. Such pattern cannot be identified using the existing algorithms for a single network. We show that even though finding the densest subgraph in a single network is polynomial time solvable, the DCS problem is NP-hard. We develop a two-step approach to solve the DCS problem. In the first step, we effectively prune the dual networks while guarantee that the optimal solution is contained in the remaining networks. For the second step, we develop two efficient greedy methods based on different search strategies to find the DCS. Different variations of the DCS problem are also studied. We perform extensive experiments on a variety of real and synthetic dual networks to evaluate the effectiveness and efficiency of the developed methods.**

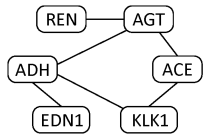
I. Introduction

Finding the densest subgraph is an important problem with a wide range of applications [1], [2]. Most of the existing work focuses on a single network, i.e., given a graph $G(V, E)$, find the subgraph with maximum density (average edge weight) [3]. This problem can be solved in polynomial time [4].

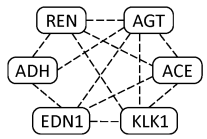
In many real-life applications, we often observe two complementary networks: one represents the *physical* interaction among a set of nodes and another one represents the *conceptual* interaction. In such applications, it is important to find subgraphs that are dense in the conceptual network and also connected in the physical network.

For example, in genetics, it is crucial to interpret genetic interactions by protein interactions. The genetic interaction network represents the conceptual interaction among genes, where the interactions are measured by statistical test [5]. Two genes with strong genetic interaction may not have physical interaction. The protein interaction network represents physical interactions and can be used to uncover the biological mechanisms behind the genetic interactions [6].

Figure 1 shows an example of the dual biological networks. Figure 1(a) shows the protein interaction network and Figure 1(b) shows the genetic interaction network[2]. The set of genes have high density in the genetic interaction network indicating their statistical interactions are strong. They are also connected in the protein interaction network, which represents the physical signal transduction pathway [7].

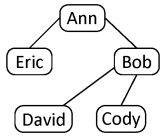


(a) Protein interaction network

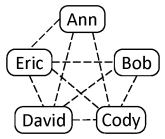


(b) Genetic interaction network

Fig. 1. Dual protein and genetic interaction networks



(a) Physical connectivity network



(b) Interest similarity network

Fig. 2. Dual user interest similarity and connectivity networks

As another example, consider the dual networks shown in Figure 2. A conceptual user similarity network can be easily derived from the user-item rating matrix (e.g., by measuring the correlation of common ratings between two users). Two users with similar interests may not have direct physical contact. However, if a set of users with highly similar interest, as shown in Figure 2(b), are also physically connected, as shown in Figure 2(a), it may be utilized for verbal recommendation. If one of the users receives an advertisement of an interested product, this information is likely to propagate to the rest of the group because of their common interest and physical connectivity.

Research interest similarity and collaboration network among researchers is another example of dual networks. The research interest network is conceptual (which can be constructed, for example, by measuring the similarity of keywords in the papers of different researchers). Two researchers with similar interest may not collaborate with each other. The collaboration network represents the physical interaction between researchers, i.e., whether two researchers have co-authored a paper.

[1]In this paper, we use network and graph interchangeably.

[2]We use solid (dotted) lines to represent the edges in the physical (conceptual) network throughout the paper.

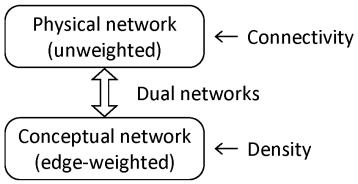


Fig. 3. DCS in dual networks

In this paper, we study the problem of finding the densest connected subgraph (DCS) in dual networks. Given two graphs $G_a(V, E_a)$ and $G_b(V, E_b)$ representing the physical and conceptual networks respectively, the DCS consists of a subset of nodes $S \subseteq V$ such that the induced subgraph $G_a[S]$ is connected and the density of $G_b[S]$ is maximized. Figure 3 summarizes the problem setting.

Note that our problem is different from finding co-dense subgraphs [8], [9] or coherent dense subgraphs [10], [11], whose goal is to find the dense subgraphs preserved across multiple networks of the same type. In our problem, the physical and conceptual networks represent complementary information and need different treatments.

Even though finding the densest subgraph in a single network can be solved in polynomial time, finding the DCS in dual networks is NP-hard. We devise a two-step approach to solve this problem. In the first step, we show that by removing low degree leaf nodes in the dual network, we can not only dramatically reduce the search space, but also guarantee the optimal solution is still retained in the remaining networks. In the second step, we develop two greedy approaches to find the DCS in the pruned dual networks. The first approach finds the densest subgraph in the conceptual network first and then refines and makes it connected in the physical network. The second approach keeps the target subgraph connected in the physical network while deleting low degree nodes in the conceptual network. We further study different variations of the DCS problem. One problem aims to find the DCS with a fixed number of nodes. Another problem requires a set of input nodes to be included in the identified subgraph. We perform extensive empirical study using a variety of real and synthetic dual networks to demonstrate the effectiveness and efficiency of the developed algorithms.

II. Related Work

Finding the densest subgraph in a single graph has attracted intensive research interests. In its basic form, the problem is to find the subgraph with maximum average edge weight. This problem can be solved in polynomial time using parametric maximum flow [4]. However, its complexity $O(nm\log(n^2/m))$ is prohibitive for large graphs, where n is the number of nodes and m is the number of edges. For large graphs, efficient approximation algorithms have been developed. A 2-approximation algorithm is proposed in [12], [13]. The basic strategy is deleting the node with minimum degree. This idea can be traced back to [14], which shows that the density of the maximum core of a graph is at least half of the density of the densest subgraph. Recently, an improved $2(1+\epsilon)$ approximation greedy node deletion algorithm has been proposed [15]. The algorithm takes $O(\log_{1+\epsilon} n)$ iterations. In each iteration, it deletes a set of nodes with degree smaller than $2(1+\epsilon)$ times the density of the remaining subgraph.

Variations of the densest subgraph problem have also been studied. The densest k subgraph problem aims to find the densest subgraph with exactly k nodes, which has been shown to be NP-hard [16]. The problem of finding the densest subgraph with seed nodes requires that a set of input nodes must be included in the resulting subgraph, which can be solved in polynomial time [1].

In computational biology, the densest subgraph has been used to analyze gene annotation graph [1]. The idea can be generalized to analyze multiple networks. For example, in [10], [11], the authors aim to find coherent dense subgraphs whose edges are not only densely connected but also frequently occur in multiple gene co-expression networks. Finding co-dense subgraphs that exist in multiple gene co-expression or protein interaction networks are studied in [8], [9]. The underlying assumption of these works is that the set of networks under study are of the same type.

Dense subgraphs have also been used to identify communities of common interests in social networks [2], [17]. In recommender systems, it has been shown that trust relation among users and their social network have the potential to further improve the performance of the algorithms [18], [19].

TABLE I
Main symbols

Symbol	Definition
$G(V, E)$	graph G with node set V and edge set E
$G(V, E_a, E_b)$	dual networks $G_a(V, E_a)$ and $G_b(V, E_b)$
S	node set $S \subseteq V$
$G[S]$	subgraph induced by S in graph G
$G_a[S], G_b[S]$	subgraph induced by S in graph G_a, G_b
$\|S\|$	number of nodes in S
$w(u, v)$	weight of edge (u, v)
$E(S)$	edge set $\{(u, v) \| u, v \in S\}$
$\|E(S)\|$	sum of edge weight $\sum_{(u,v)\in E(S)} w(u, v)$
$\rho(S)$	density (average edge weight) of subgraph $G[S]$
$w_G(u)$	degree of node u in graph G

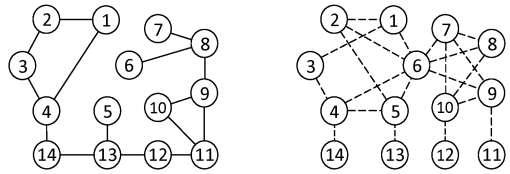
III. The DCS Problem

We adopt the classic graph density definition [12], [15], [13], [4] to formulate the DCS problem. Table I lists the main symbols and their definitions.

Definition 1: Given a graph $G(V, E)$ and $S \subseteq V$, density $\rho(S)$ is defined as

$$\rho(S) = \frac{|E(S)|}{|S|}.$$

Let $G_a(V, E_a)$ be an unweighted graph representing the physical network and $G_b(V, E_b)$ be an edge weighted graph representing the conceptual network. We denote the subgraphs



(a) Physical network G_a (b) Conceptual network G_b

Fig. 4. An example of dual networks

induced by node set $S \subseteq V$ in the physical and conceptual networks as $G_a[S]$ and $G_b[S]$ respectively, and edge sets induced by S as $E_a(S)$ and $E_b(S)$ respectively. For brevity, we also use $G(V, E_a, E_b)$ to represent the dual networks.

Definition 2: Given dual networks $G(V, E_a, E_b)$, the densest connected subgraph (DCS) consists of a set of nodes $S \subseteq V$ such that $G_a[S]$ is connected and the density of $G_b[S]$ is maximized.

An example is shown in Figure 4. In this example, the DCS consists of nodes $S = \{6, 7, 8, 9, 10\}$. Its induced subgraph $G_a[S]$ is connected in the physical network and $G_b[S]$ has the largest density in the conceptual network. Note that the dense component consisting of nodes $\{1, 2, 3, 4, 5, 6\}$ in G_b is not connected in G_a.

Theorem 1: Finding the DCS in dual networks is NP-hard.

Proof: We show that the DCS problem can be reduced from the set cover problem [20]. Let $C = \{C_1, \cdots, C_q\}$ be a family of sets with $R = \{r_1, \cdots, r_p\} = \bigcup_{i=1}^{q} C_i$ being the elements. The set cover problem aims to find a minimum subset $C_{opt} \subseteq C$, such that each element r_j is contained in at least one set in C_{opt}.

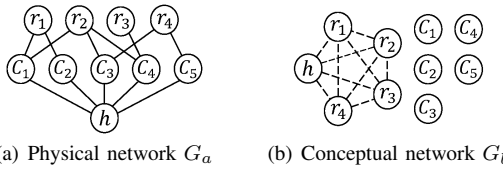
The dual networks can be constructed as follows. Let the node set $V = \{h, r_1, \cdots, r_p, C_1, \cdots, C_q\}$. In the physical network G_a, node h is connected to every node $C_i \in C$, and every node $r_j \in R$ is connected to node C_i if $r_j \in C_i$ in the set cover problem. The conceptual network G_b is constructed by creating a unit edge weight clique among nodes $\{h, r_1, \cdots, r_p\}$ and leaving nodes $\{C_1, \cdots, C_q\}$ isolated.

Figure 5 gives an example of the dual networks constructed from an instance of the set cover problem with $C_1 = \{r_1, r_2\}, C_2 = \{r_1\}, C_3 = \{r_2, r_4\}, C_4 = \{r_2, r_3\}, C_5 = \{r_4\}$.

Let $C_{opt} \subseteq C$ be the optimal solution to the set cover problem and $|C_{opt}| = q^* \leq q$. Denote $H = \{h, r_1, \cdots, r_p\}$. The subgraph induced from $S = H \cup C_{opt}$ is connected in G_a, and has density $\frac{p(p+1)/2}{p+q^*+1}$ in G_b. Let S' denote any node set, where $G_a[S']$ is connected. Next, we prove that the density of $G_b[S]$ is no less than that of $G_b[S']$.

First, we consider the case when S' contains all nodes in H. S' must contain a set of nodes $C' \subseteq C$ to be connected in G_a. Thus $S' = H \cup C'$, $|E_b(S')| = p(p+1)/2$, and $|S'| = p + 1 + |C'|$. Since C_{opt} has the minimum number of sets (nodes) among all subsets of C that cover all elements in R, the density of $G_b[S]$ is no less than that of $G_b[S']$.

Second, we consider the case when S' contains a subset of nodes $H' \subset H$. S' must contain a set of nodes $C' \subseteq C$ to be connected in G_a. Thus $S' = H' \cup C'$. Let $|H'| = p'$ and



(a) Physical network G_a (b) Conceptual network G_b

Fig. 5. Dual networks construction from an instance of the set cover problem

$|C'| = q' \geq 1$. The density of $G_b[S']$ is $\frac{p'(p'-1)/2}{p'+q'}$. Next, we show that adding nodes in $H \setminus H'$ to S' will only increase its density.

If $h \notin S'$, after adding h to S', the resulting subgraph has density $\frac{p'(p'-1)/2+p'}{p'+q'+1} > \frac{p'(p'-1)/2}{p'+q'}$ in G_b, and is also connected in G_a since h is connected to every $C_i \in C$. To add a node $r_j \in H \setminus H'$ to S' and make it still connected, we need to add at most one node C_i, where $r_j \in C_i$. The density of the resulting subgraph is at least $\frac{p'(p'-1)/2+p'}{p'+q'+2} > \frac{p'(p'-1)/2}{p'+q'}$. We can repeat this process by adding remaining nodes to S' until it contains all the nodes in H. During this process, the density of the resulting subgraph will keep increasing. In the first case, we already prove that the density of $G_b[S]$ is no less than that of $G_b[S']$ when $H \subset S'$. This completes the proof for the second case.

Therefore, the subgraph induced from $S = H \cup C_{opt}$ is the DCS, and it gives an optimal solution to the set cover problem.

Let's continue the example in Figure 5. The subgraph induced from $S = \{h, r_1, r_2, r_3, r_4, C_1, C_3, C_4\}$ is the DCS which is connected in G_a and has maximum density 1.25 in G_b. $C_{opt} = \{C_1, C_3, C_4\}$ is an optimal solution to the set cover problem. ■

The DCS with size constraint (DCS_k) and input seed nodes (DCS_*seed*) can be defined as follows.

Definition 3: Given dual networks $G(V, E_a, E_b)$ and an integer k, the DCS_k consists of a set of nodes $S \subseteq V$ such that $|S| = k$, $G_a[S]$ is connected and the density of $G_b[S]$ is maximized.

Definition 4: Given dual networks $G(V, E_a, E_b)$ and an input node set $U \subseteq V$, the DCS_*seed* consists of a set of nodes $S \subseteq V$ such that $U \subseteq S$, $G_a[S]$ is connected and the density of $G_b[S]$ is maximized.

The DCS_k and DCS_*seed* problems are also NP-hard. The proofs are omitted.

IV. Optimality Preserving Pruning

In this section, we introduce a pruning step, which removes the *low degree leaf nodes* from the dual networks and still guarantees that the optimal DCS is contained in the resulting networks.

Definition 5: Given dual networks $G(V, E_a, E_b)$, suppose that its DCS consists of a set of nodes S. Let $\rho(S)$ represent its density in G_b, i.e., $\rho(S) = \rho(G_b[S])$. A node $u \in V$ is a low degree leaf node if (1) u is a leaf node in G_a, i.e., $w_{G_a}(u) = 1$, and (2) its degree in G_b is less than $\rho(S)$, i.e., $w_{G_b}(u) < \rho(S)$.

Lemma 1: The DCS in dual networks does not contain any low degree leaf node.

Proof: Suppose otherwise. We remove u from S and let S' be the remaining set of nodes. Since $G_a[S]$ is connected and u is a leaf node in G_a, so after deleting u, $G_a[S']$ is still connected. However, its density $\rho(S') = \frac{|E_b(S')|}{|S'|} = \frac{|E_b(S)|-w_{G_b}(u)}{|S|-1} > \frac{|E_b(S)|}{|S|} = \rho(S)$, since $w_{G_b}(u) < \rho(S) = \frac{|E_b(S)|}{|S|}$. This contradicts the assumption. ∎

Even though the density of DCS ($\rho(S)$) is unknown beforehand, we can still effectively prune many low degree leaf nodes as follows. Let $G_0 = G$ be the original dual networks. We remove all low degree leaf nodes (using density $\rho(G_b[V])$) in the physical network G_0^a and conceptual network G_0^b, respectively. That is, we remove all the nodes that have degree one in G_a and have degree less than $\rho(G_b[V])$ in G_b from the dual networks. Let the resulting dual networks be $G_1(V_1, E_a(V_1), E_b(V_1))$. We then continue to remove the low degree leaf nodes using density $\rho(V_1)$ in G_1. That is, we remove all the nodes that have degree one in $G_a[V_1]$ and have degree less than $\rho(G_b[V_1])$ in G_b from the dual networks. We repeat this process until no such nodes left.

Let $\{G_0, G_1, \cdots, G_l\}$ represent the sequence of dual networks generated by this process and $\{v_j^i\}$ represent the set of nodes deleted in iteration i ($0 \leq i \leq l$). The following theorem shows that the DCS is retained in this process.

Theorem 2: Iteratively removing low degree leaf nodes will not delete any node in the DCS.

Proof: Consider two adjacent dual networks G_i and G_{i+1} in the sequence $\{G_0, G_1, \cdots, G_l\}$. From G_i to G_{i+1}, we delete a set of nodes $\{v_j^i\}$. For a node $u \in \{v_j^i\}$, it is a leaf node in G_i^a, and its degree in G_i^b is $w_{G_i^b}(u) < \rho(G_i^b)$. Let S_i be the node set of the DCS in G_i. We have that $\rho(G_i^b) \leq \rho(S_i)$. Thus $w_{G_i^b}(u) < \rho(S_i)$. Therefore, node u is a low degree leaf node with respect to the DCS in G_i. From the proof of Lemma 1, node u must not exist in S_i. By induction, we have that the DCS of the original dual networks is retained in the low degree leaf nodes removing process. ∎

Using this pruning strategy, we can safely remove the nodes that are not in the DCS, thus reduce the overall search space. Experimental results on real graphs show that 40% to 60% of the nodes can be pruned using this method.

Next, we introduce two greedy algorithms, DCS_RDS (Refining the Densest Subgraph) and DCS_GND (Greedy Node Deletion), to find the DCS from the size reduced dual networks.

V. The DCS_RDS Algorithm

The DCS_RDS algorithm first finds the densest subgraph in G_b, which usually is disconnected in G_a. It then refines the subgraph by connecting its disconnected components in G_a. Although the densest subgraph can be identified in polynomial time by the parametric maximum flow method [4], its actual complexity $O(nm\log(n^2/m))$ is prohibitive for large graphs (n and m are the number of nodes and edges in the graph respectively). Next, we first introduce an effective procedure that can dramatically reduce the cost of finding the densest subgraph in a single graph.

A. Fast Densest Subgraph Finding in Conceptual Network

To find the densest subgraph in a single network, greedy node deletion algorithms [12], [13] and peeling algorithms [21], [15], [22] keep deleting the nodes with low degree. However, these methods do not guarantee that the densest subgraph is contained in the identified subgraph.

We introduce an approach which effectively removes nodes in G_b and still guarantees to retain the densest subgraph. Our node removal procedure is based on the following key observation.

Lemma 2: Let $\rho(T)$ be the density of the densest subgraph $G[T]$. Any node $u \in T$ has degree $w_{G[T]}(u) \geq \rho(T)$.

Proof: Suppose there exists a node $u \in T$ with $w_{G[T]}(u) < \rho(T)$. Then the subgraph $G[T'] = G[T \setminus \{u\}]$ has density $\rho(T') = \frac{|E(T)|-w_{G[T]}(u)}{|T|-1} > \frac{|E(T)|}{|T|} = \rho(T)$. Thus we find a subgraph $G[T']$ whose density is larger than that of $G[T]$. This contradicts the assumption that $G[T]$ is the densest subgraph. ∎

The lemma says that the degree of any node in the densest subgraph $G[T]$ must be no less than its density $\rho(T)$. Since the density of $G[T]$ is also equivalent to half of the average degree in $G[T]$, i.e., $\rho(T) = \overline{w}_{G[T]}/2$, this is equivalent to say that any node should have degree more than $\overline{w}_{G[T]}/2$. Note that this is a necessary condition for characterizing the densest subgraph. It is also related to the concept of d-core.

Definition 6: The d-core D of G is the maximal subgraph of G such that for any node u in D, $w_D(u) \geq d$.

Note that the d-core of a graph is unique and may consist of multiple connected components. It is easy to see that any subgraph in which every node's degree is no less than d is part of the d-core.

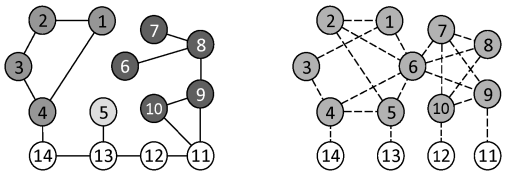
Theorem 3: The densest subgraph $G[T]$ of G is a subgraph of the d-core D of G ($G[T] \subseteq D$) when $d \leq \rho(T)$.

Proof: From Lemma 2, any node $u \in T$ has degree $w_{G[T]}(u) \geq \rho(T)$. Since $d \leq \rho(T)$, any node $u \in T$ has degree $w_{G[T]}(u) \geq d$. Thus $G[T]$ is a subgraph of the d-core D. ∎

Lemma 3: Let $\alpha = \rho(T)/d$. The d-core subgraph D is a 2α-approximation of the densest subgraph $G[T]$.

Proof: Let $D.V$ represent the node set in D. Since the density of the d-core is $\rho(D) = \frac{|E(D.V)|}{|D.V|} = \frac{\sum_{u\in D.V} w_D(u)}{2|D.V|} \geq \frac{\sum_{u\in D.V} d}{2|D.V|} = \frac{d}{2} = \frac{\rho(T)}{2\alpha}$, we have that $\rho(T) \leq 2\alpha\rho(D)$. ∎

From Theorem 3 and Lemma 3, if we can find a density value d ($d \leq \rho(T)$), then we have both: 1) $G[T] \subseteq D$, and 2) D is a $2\rho(T)/d$ approximation of the densest subgraph $G[T]$. Therefore, if we use the density of the 2-approximation subgraph generated by the greedy node deletion algorithm [12], [13] for d-core, we obtain a 4-approximation ratio



(a) Physical network G_a (b) Conceptual network G_b

Fig. 6. Refining the densest subgraph

$(2\alpha \leq 2(2d)/d = 4)$ of the densest subgraph $G[T]$. Note that d-core can be generated by iteratively removing all nodes with degree less than d until every node in the remaining graph has degree no less than d [15].

To sum up, we use the following three-step procedure to find the exact densest subgraph from G: (1) Find a 2-approximation of the densest subgraph in G, where the density of the discovered subgraph $d \geq \rho(T)/2$; (2) Find the d-core D of G; (3) Compute the exact densest subgraph from D.

Empirical results show that after applying this approach, the remaining subgraph can be orders of magnitude smaller than the original graphs. It can significantly speed up the process of finding the exact densest subgraph. Moreover, we have shown that the density of the remaining subgraph is a 4-approximation of the density of the densest subgraph.

Complexity: Let n and m be the number of nodes and edges in the original graph G_b, and n' and m' be the number of nodes and edges in the d-core D. The first and second steps run in $O(m + n \log n)$ and $O(m)$ respectively. To find the exact densest subgraph from D, the parametric maximum flow algorithm runs in $O(n'm' \log(n'^2/m'))$. Note that n' (m') can be orders of magnitude smaller than n (m).

B. Refining Subgraph in Physical Network

Suppose that the densest subgraph of G_b consists of node set T and is denoted as $G_b[T]$. The induced subgraph in the physical network $G_a[T]$ is typically disconnected. Given dual networks $G(V, E_a, E_b)$ and the densest subgraph $G_b[T]$, we use $\{G_a[V_1], G_a[V_2], \cdots, G_a[V_t]\}$ to represent all connected components in $G_a[T]$, where $T = V_1 \cup V_2 \cup \cdots \cup V_t$.

Example 1: In Figure 6, the densest subgraph in the conceptual network consists of nodes $T = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. Its corresponding connected components in the physical network are $V_1 = \{6, 7, 8, 9, 10\}$, $V_2 = \{1, 2, 3, 4\}$, and $V_3 = \{5\}$.

In the next, we discuss how to refine the subgraph $G_a[T]$ to make it connected in the physical network G_a while still preserving its high density in G_b. Specifically, we consider the following dense subgraph refinement problem.

Definition 7: Given dual networks $G(V, E_a, E_b)$ and the densest subgraph $G_b[T]$ of G_b, the problem of refining the densest subgraph aims to find a nonempty subset of $\{G_a[V_1], G_a[V_2], \cdots, G_a[V_t]\}$ with node set Y and a node set $X \subseteq V \setminus T$, such that $G_a[Y \cup X]$ is connected and the density of $G_b[Y \cup X]$ is maximized.

The problem of refining the densest subgraph is also NP-hard, which can be proved using similar reduction method as in the proof of Theorem 1.

Algorithm 1: Refining the densest subgraph

Input: $G(V, E_a, E_b)$, nodes T (densest subgraph in G_b)
Output: node set $\hat{S}$ of DCS

1: Find all t connected components $\{G_a[V_i]\}$ in $G_a[T]$;
2: Sort $G_a[V_i]$ by density $\rho(G_b[V_i])$ in descending order;
3: Weigh the node u in G_a by $(w_{G_b}(u))^{-1}$;
4: $S_1 \leftarrow V_1$;
5: **for** $i \leftarrow 1$ *to* $t-1$ **do**
6: Compute shortest path $H_i(S_i, V_{i+1})$ in G_a;
7: $S_{i+1} \leftarrow S_i \cup V_{i+1} \cup H_i$;
8: $j \leftarrow \text{argmax}_i\, \rho(G_b[S_i])$; **return** S_j;

We introduce a greedy heuristic procedure to refine the densest subgraph as outlined in Algorithm 1. The algorithm puts the node set T to G_a, and finds all the connected components $\{G_a[V_i]\}$ in line 1. It then sorts $\{G_a[V_i]\}$ by their density $\rho(G_b[V_i])$ in descending order in line 2. In line 3, it weights the nodes in G_a by the reciprocal of its degree in G_b. The intuition is that we want to select nodes that have high degree in G_b to connect $\{G_a[V_i]\}$. The algorithm merges the connected components in G_a iteratively. In each iteration in lines 6-7, it merges two components by adding the nodes on the node weighted shortest path connecting two components. The density of the newly merged component is calculated after each iteration. The component with the largest density is returned as the DCS.

Example 2: Continue the example in Figure 6. The densities of the connected components in the physical network are $\rho(V_1 = \{6, 7, 8, 9, 10\}) = 1.6$, $\rho(V_2 = \{1, 2, 3, 4\}) = 0.75$, and $\rho(V_3 = \{5\}) = 0$. Initially, the subgraph induced by $S_1 = V_1$ has density $\rho(S_1) = 1.6$. Algorithm 1 first connects S_1 and V_2 through the shortest path $H_1 = \{11, 12, 13, 14\}$. The subgraph induced by $S_2 = S_1 \cup V_2 \cup H_1$ has density $\rho(S_2) = 1.31$. After merging V_3, the subgraph induced by S_3 has density $\rho(S_3) = 1.5$. Therefore, the subgraph induced by S_1 has the largest density in G_b and is returned as the DCS.

The approximation ratio of the DCS_RDS algorithm can be estimated as $\alpha = \rho(T)/\rho(\hat{S})$, where $\rho(T)$ is the density of the densest subgraph in the conceptual network. Experimental results show that the approximation ratio is usually around 1.5∼2 using real networks.

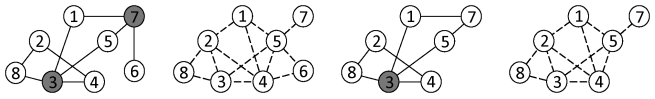
Complexity: Algorithm 1 runs in $O(m + n \log n)$ as we can easily modify Dijkstra's algorithm to find the shortest path in node weighted graph by transforming each node as an edge.

VI. The DCS_GND Algorithm

The basic DCS_GND algorithm keeps deleting nodes with low degree in the conceptual network, while avoiding disconnecting the physical network.

Definition 8: A node is an articulation node if removing this node and the edges incident to it disconnects the graph.

Articulation nodes can be identified in linear time by depth first search [23]. The basic DCS_GND algorithm deletes one



(a) G_0^a (b) G_0^b (c) G_1^a (d) G_1^b

Fig. 7. Greedy node deletion example

node in each iteration. The deleted node has the minimum degree in the conceptual network among all the non-articulation nodes in the physical network. Since in each iteration, only one non-articulation node is deleted, the remaining physical network will keep connected. Note that as long as the graph is not empty, there always exists a non-articulation node in the graph. Thus, the DCS_GND algorithm can always find a non-articulation node to delete until the graph becomes empty. Density of the subgraphs generated in this process is recorded and the subgraph with the largest density is returned as the identified DCS.

Example 3: Suppose that the input physical and conceptual networks are as shown in Figures 7(a) and 7(b) respectively. Nodes $\{3, 7\}$ in gray color are articulation nodes and the remaining ones are non-articulation nodes. Node 6, which has the minimum degree 2 among all the non-articulation nodes, will be deleted. The resulting dual networks are shown in Figures 7(c) and 7(d), where node 3 is the only articulation node.

To further improve the efficiency, we can delete a set of low degree non-articulation nodes in each iteration. However, not all non-articulation nodes can be deleted simultaneously, since deleting one non-articulation node may make another non-articulation node to become an articulation node. Thus we need to find the subset of non-articulation nodes that can be deleted together.

Definition 9: A set of non-articulation nodes are independent if the deletion of them does not disconnect the graph.

Lemma 4: Let $\{B_i\}$ represent the set of biconnected components of graph G such that each B_i has at least one non-articulation node. If we select one non-articulation node from each B_i, the set of selected nodes are independent non-articulation nodes.

Proof: Suppose that we delete one non-articulation node $v_i \in B_i$. The deletion of node v_i does not disconnect B_i since it is biconnected. Since two distinct biconnected components share at most one articulation node, deleting node v_i does not disconnect any other biconnected components. So if we delete one non-articulation node from each component in $\{B_i\}$, every component is still connected. Therefore, the remaining subgraph is still connected. ■

Algorithm 2 illustrates the algorithm based on deleting independent non-articulation nodes iteratively. Parameter γ in line 4 is used to control the degree of the non-articulation nodes to be deleted. γ is usually set between $0 \sim 2$. Since $2\rho(G_b)$ is the average node degree, there are about half of the nodes whose degree is smaller than the threshold $2\rho(G_b)$.

Algorithm 2: Fast DCS_GND algorithm

Input: $G(V, E_a, E_b)$, parameter $\gamma > 0$
Output: node set $\hat{S}$ of DCS

1: $V_0 \leftarrow V$; $i \leftarrow 0$;
2: **while** $|V_i| > 0$ **do**
3: Compute the articulation nodes A in $G_a[V_i]$; $A \leftarrow V_i \setminus A$;
4: Select a set of nodes $L \subseteq A$ such that the nodes in L are independent non-articulation nodes and have low degrees, i.e., for any node $u \in L$, $w_{G_b[V_i]}(u) \leq \gamma \cdot \rho(G_b[V_i])$;
5: **if** $|L| = 0$ **then** $L \leftarrow \{u \,|\, u = \text{argmin}_{v \in A}\, w_{G_b[V_i]}(v)\}$;
6: $V_{i+1} \leftarrow V_i \setminus L$; $i \leftarrow i + 1$;
7: $j \leftarrow \text{argmax}_i\, \rho(G_b[V_i])$; **return** V_j;

More nodes are deleted in each iteration when larger γ value is used. Please refer to experimental evaluation for further discussion on the effect of γ. If all low degree nodes are articulation nodes, the algorithm picks the non-articulation node with the minimum degree to delete as shown in line 5.

Example 4: Let's continue the example in Figure 7. Suppose that $\gamma = 1.5$. We have $\gamma\rho(G_b[V_0]) = 2.44$. The fast DCS_GND method will delete nodes $\{6, 8\}$ simultaneously, since they have degree $2 < 2.44$ and are independent non-articulation nodes.

Complexity: The DCS_GND algorithm runs in $O(nm)$ for finding the articulation nodes and biconnected components by depth first search.

We can estimate the approximation ratio of DCS_GND as follows. When deleting a node v, we assign its incident edges in G_b to it. Let $edg(v)$ denote the sum of the edge weights, and edg_{max} represent the maximum $edg(v)$ among all nodes (deleted in order by the algorithm). Let S be the node set of the optimal DCS, T be the node set of the densest subgraph in the conceptual network G_b. We have the following inequality.

Lemma 5: $\rho(S) \leq \rho(T) \leq edg_{max}$.

Proof: It is easy to see that $\rho(S) \leq \rho(T)$. Next, we show $\rho(T) \leq edg_{max}$. Each edge in $E_b(T)$ must be assigned to a node in T in the node deletion process. Thus we have that $|E_b(T)| \leq \sum_{u \in T} edg(u) \leq \sum_{u \in T} edg_{max} = |T| \cdot edg_{max}$. This means $\rho(T) = \frac{|E_b(T)|}{|T|} \leq edg_{max}$. ■

Lemma 6: The approximation ratio of the DCS_GND algorithm is $\alpha = edg_{max}/\rho(\hat{S}) \geq \rho(S)/\rho(\hat{S})$, where $\hat{S}$ is the node set identified by the algorithm.

Based on Lemma 6, we can estimate the approximation ratio α from the results returned by the algorithm. Empirical study shows that α is usually around 2 in real networks.

The DCS_GND algorithm can be easily extended to solve the DCS_k and DCS_*seed* problems. For the DCS_k problem, we can keep deleting low degree non-articulation nodes until there are k nodes left. For the DCS_*seed* problem, we avoid deleting the seed nodes during the process. The approximation ratio analysis discussed above also applies to these variants.

TABLE II
STATISTICS OF THE DUAL NETWORKS

Dual networks	Abbr.	#nodes	#edges in G_a	#edges in G_b
Research-DM	DM	$7,169$	$14,526$	$30,000$
Research-DB	DB	$6,131$	$17,940$	$30,000$
Recom-Epinions	EP	$49,288$	$487,002$	$313,432$
Recom-Flixster	FX	$786,936$	$7,058,819$	$2,713,671$
Protein-Genetic	Bio	$8,468$	$25,715$	$67,744$

VII. EXPERIMENTAL RESULTS

In this section, we perform comprehensive experiments to evaluate the effectiveness and efficiency of the proposed methods using a variety of real and synthetic datasets. All the programs are written in C++. All experiments are performed on a server with 32G memory, Intel Xeon 3.2GHz CPU, and Redhat OS.

A. Real Dual Networks

We constructed several dual networks from different application domains. We use the DBLP dataset [24] to build two dual networks, one for data mining research community and one for database research community. To construct the dual networks for the data mining community, we extract a set of papers published in 5 data mining conferences: KDD, ICDM, SDM, PKDD and CIKM. The dataset contains 4,284 papers and 7,169 authors. The physical network is the co-author network with authors being the nodes and edges representing two authors have co-authored a paper. The conceptual research interest similarity network among authors is constructed based on the similarity of the terms in the paper titles of different authors. The shrunk Pearson correlation coefficient is used to compute the research interest similarity between authors [25]. The dual networks for the database community are constructed in a similar way based on papers published in SIGMOD, VLDB and ICDE.

We construct two dual networks using recommender system datasets, Flixster [26] and Epinions [27]. In the original Flixster dataset, the physical network has 786,936 nodes (users) and 7,058,819 edges representing their social connectivity. The user-item rating matrix consists of 8,184,462 user ratings for 48,791 items with rating scale from 1 to 5 with 0.5 increment. We construct the conceptual interest similarity network by measuring the correlation coefficients of the common ratings between users [19]. Note that we only calculate the correlation coefficient between two users with more than 5 common ratings. The constructed interest similarity network has 2,713,671 edges. The trust network in Epinions dataset has 49,288 nodes and 487,002 edges. The user-item rating matrix consists of 664,811 user ratings for 139,737 items with rating scale from 1 to 5 with 1 increment. The interest similarity network is constructed in a similar way as the one in the Flixster dataset. It has 313,432 edges.

The biological dual networks include the physical protein interaction network and the conceptual genetic interaction network. The protein interaction network is downloaded from the BioGRID database (*http://thebiogrid.org/*). After filtering out duplicate interactions, the network contains 8,468 proteins and 25,715 unique physical bonding interactions. The genetic interaction network is generated by performing chi-square test on genetic marker pairs in the Wellcome Trust Case Control Consortium (WTCCC) hypertension dataset [28]. The most significant interactions between genes are used to weight the edges in the genetic interaction network, which has 67,744 edges. Note that we use half of the samples in the WTCCC dataset to construct the dual networks. Another half is used for significance evaluation of the identified DCS.

Table II shows the basic statistics of the real dual networks used in the experiments.

B. Effectiveness Evaluation

1) Research interest similarity and co-author dual networks: The DCS identified in the dual co-author networks consists of hundreds of nodes. Here we only show the identified DCS_k for data mining in Figure 8 and DCS_*seed* for database in Figure 9.

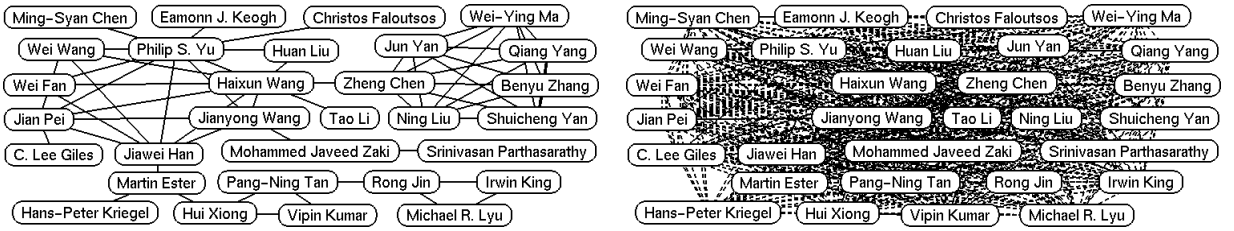
Figures 8(a) and 8(b) shows the DCS_k ($k = 30$) identified in the dual networks of the data mining research community. The subgraph in the co-author network is sparsely connected and highly dense in the research interest similarity network. This indicates that the set of researchers have very close research interest. The subgraph in the co-author network shows their collaboration pattern.

Figures 9(a) and 9(b) shows the DCS_*seed* identified in the dual networks of the database research community. The names of the 4 input seed authors are in red ellipses. The researchers with similar interest and their collaboration patterns are clearly shown in the two subgraphs. The 4 seed authors do not have direct co-authorship with each other. Through the resulting DCS_*seed*, we uncover the connected community of common interests.

Note that a dense subgraph in the research interest similarity network may not be connected in the co-author network. One example is shown in Figure 10. Figure 10(b) shows a dense subgraph identified from the research interest similarity network for data mining. Figure 10(a) shows the induced subgraph in the co-author network. We can see that very few authors are connected. Thus finding dense subgraphs in a single network may miss important information presented in the other network.

2) User interest similarity and social connectivity dual networks: The DCS_k ($k = 40$) identified in the dual network constructed from the Epinions dataset is shown in Figure 11. The subgraph in the interest similarity network is a dense component and not shown here. In this figure, each node is a user, whose name is not shown because of the privacy issue. Because this group of users have high interest similarity and also have social connection, if one of the users receives an advertisement of an interested product, this information is likely to be propagated to the rest of the group.

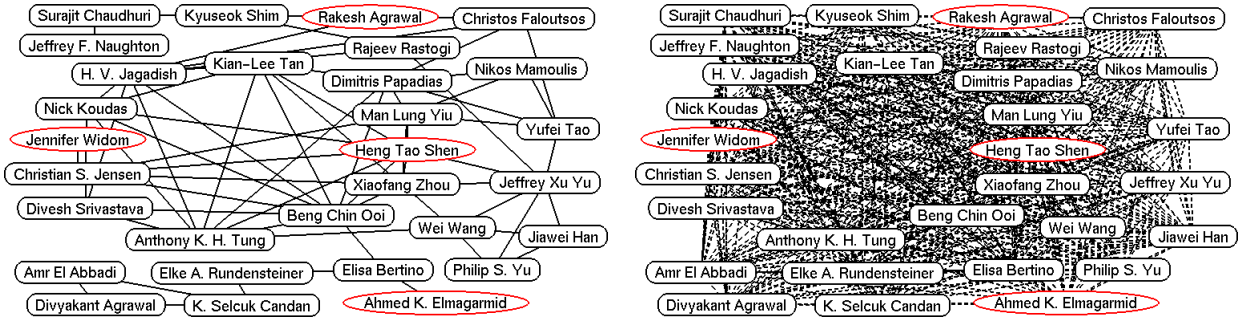
To demonstrate the effectiveness of the DCS pattern, we compare it with the dense subgraphs discovered from a single



(a) Subgraph in co-author network

(b) Subgraph in research interest similarity network

Fig. 8. The DCS_k ($k = 30$) identified from the dual co-author (data mining) networks



(a) Subgraph in co-author network

(b) Subgraph in research interest similarity network

Fig. 9. The DCS_*seed* identified from the dual co-author (database) networks

(a) Induced subgraph in co-author network

(b) Dense subgraph in research interest similarity network

Fig. 10. The dense subgraph in the research interest similarity network of the dual co-author (data mining) networks

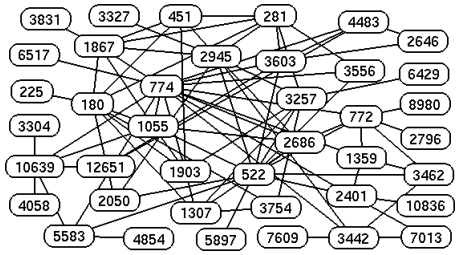


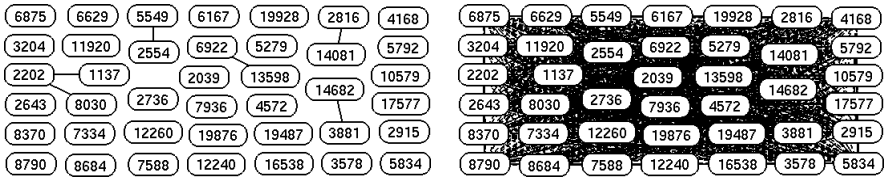
Fig. 11. The social connectivity network of the DCS_k ($k = 40$) identified in the Epinions dataset

network. Figure 12(b) shows a dense subgraph in the interest similarity network. Figure 12(a) shows its induced subgraph in the social connectivity network. We can see that this set of users have no social connectivity even though they have high interest similarity. Figure 13(a) shows a dense subgraph in the social connectivity network. Figure 13(b) shows its induced subgraph in the interest similarity network. We can see that the subgraph in the interest similarity network is very sparse. This indicates that a group of users having high social connectivity may not have similar interest. Similar observations can be made in the dual networks constructed from the Flixster dataset.

3) Protein and genetic interaction dual networks: The DCS identified in the dual biological networks has 211 nodes. The figures are omitted because of the large size. The set of nodes are sparsely connected in the protein interaction network, while the subgraph in the genetic interaction network has high density. Specifically, the DCS has 282 edges in the protein interaction network and $4,258$ edges in the genetic interaction network.

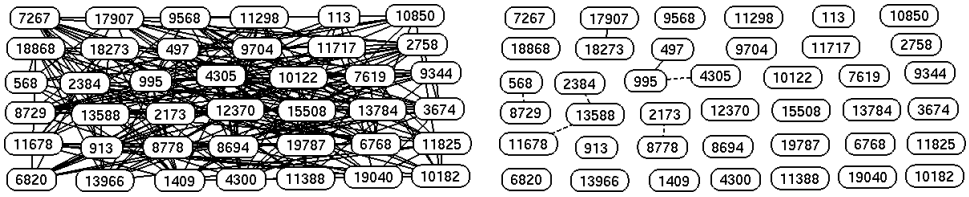
Note that the densest subgraph of the genetic interaction network is not connected in the protein interaction network. There are 73 nodes in the densest subgraph of the genetic interaction network. Only 2 of them are connected in the protein interaction network. This demonstrates that dual networks can help to uncover pattern that cannot be identified in individual networks. Such pattern cannot be identified by finding dense subgraphs preserved in both networks either. There are 68 overlapping nodes between the densest subgraph in the genetic interaction network and the DCS identified by DCS_RDS.

Figure 14 shows the identified DCS_k with $k = 40$. From the figure, it is clear that the identified subgraph is connected in the protein interaction network and highly dense in the genetic interaction network. Several genes in this subgraph have been reported to be associated with hypertension. For example,



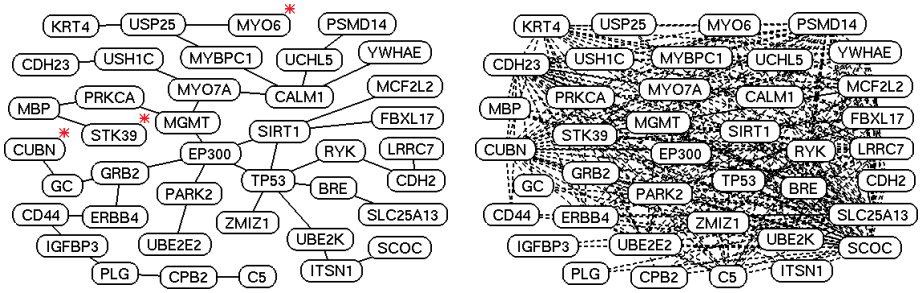
(a) Induced subgraph in social connectivity network (b) Dense subgraph in interest similarity network

Fig. 12. The dense subgraph in the interest similarity network of the Epinions dataset



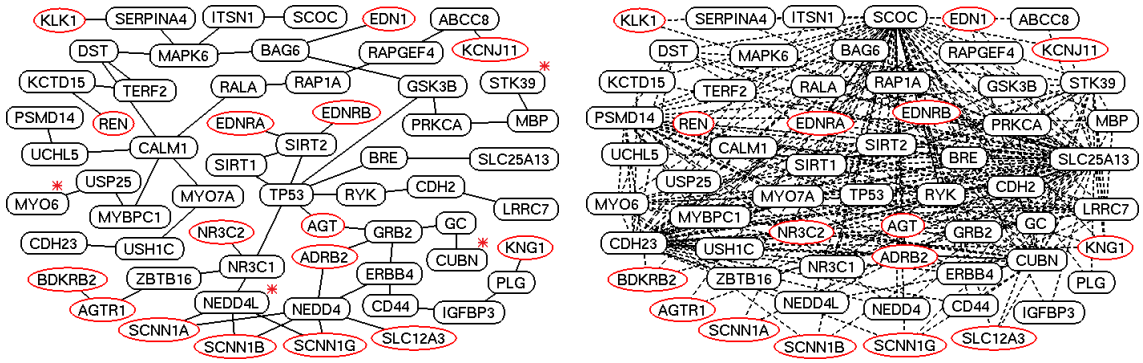
(a) Dense subgraph in social connectivity network (b) Induced subgraph in interest similarity network

Fig. 13. The dense subgraph in the social connectivity network of the Epinions dataset



(a) Subgraph in protein interaction network (b) Subgraph in genetic interaction network

Fig. 14. The DCS_k ($k = 40$) identified from the dual biological networks



(a) Subgraph in protein interaction network (b) Subgraph in genetic interaction network

Fig. 15. The DCS_*seed* identified from the dual biological networks (renin pathway genes are in red ellipses)

MYO6 encodes an actin-based molecular motor involved in intracellular vesicle and organelle transport, and has been shown to have association with hypertension [29]. The CUBN gene is associated with albuminuria, which is an important factor for cardiovascular disease [30]. The STK39 gene has been reported many times as a hypertension susceptibility gene [31]. This gene encodes a serine/threonine kinase that is thought to function in the cellular stress response pathway. These genes are highlighted by stars in the figure. Other genes in the identified subgraph are potential hypertension candidate genes or important for signal transduction in hypertension related pathways.

To identify the DCS_*seed*, we use a set of 16 genes in renin pathways known to be associated with hypertension as the input seed nodes [32]. Renin pathway, also called renin-angiotensin system, is a hormone system that regulates blood pressure. The resulting subgraphs are shown in Figure 15. The input seed genes are in red ellipses and the remaining nodes represent the newly added genes. As can be seen from the figure, the seed nodes are originally not directly connected in the protein interaction network. The newly added genes tend to have large degree in the genetic interaction network. In addition to the genes discussed above, we can see the NEDD4L gene is connected to multiple seed genes. It has

TABLE III
P-VALUE OF THE IDENTIFIED DCSS

Methods	GenGen	GRASS	Plink	HYST
DCS (1)	2.4×10^{-6}	1.0×10^{-6}	2.3×10^{-6}	1.1×10^{-9}
DCS (2)	1.6×10^{-5}	2.8×10^{-5}	4.6×10^{-5}	5.6×10^{-7}
DCS (3)	4.8×10^{-5}	7.4×10^{-5}	9.5×10^{-5}	8.2×10^{-7}
DCS_k	5.6×10^{-5}	1.3×10^{-6}	4.6×10^{-6}	3.7×10^{-8}
DCS_*seed*	8.5×10^{-5}	4.9×10^{-6}	1.5×10^{-5}	2.4×10^{-6}
DS	0.36	0.47	0.33	0.17
MSCS	0.15	0.13	0.21	0.12

been reported that NEDD4L is involved in the regulation of plasma volume and blood pressure by controlling cell surface expression of the kidney epithelial Na^+ channel [33].

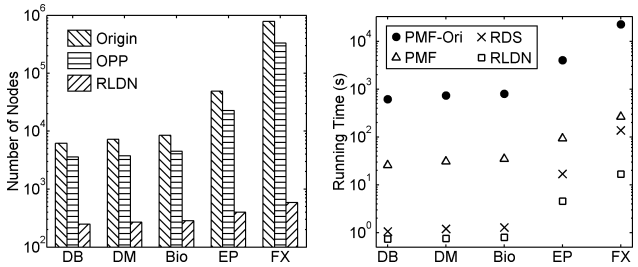
To evaluate the statistical significance of the discovered DCSs, we apply 4 widely used pathway evaluation methods : the GenGen method, gene set ridge regression (GRASS), Plink set-based test, and hybrid set-based test (HYST) [34], [35]. Given a set of genes, these methods evaluate the significance of the association between the set of genes and the disease phenotype. Note that the test dataset consists of the samples that are not used for constructing the dual networks to ensure the independence between pattern discovering and significance evaluation.

Table III shows the p-value of the identified DCSs. DCS (1), (2) and (3) represent the top-3 DCSs. The top DCSs are identified iteratively: after the top-1 DCS is identified, we remove its nodes and edges from the dual networks; the DCS_GND algorithm is then applied to each connected component to find the next DCS in the remaining graph. As can be seen from the table, the DCSs are highly significant. In the table, we also show the results of two other methods for finding pathways in biological networks. One method finds the densest subgraph (DS) in the protein interaction network. Another method aims to find the maximum-score connected subgraph (MSCS) in the protein interaction network [36]. The DS method uses the most significant genetic interactions between genes to weight the edges in the protein interaction network. The MSCS method uses the most significant chi-square test statistics to weight the nodes in the protein interaction network. As we can see, the subgraphs identified by these two methods are not as significant as the DCSs. This indicates the importance of integrating the complementary information encoded in the physical protein interaction network and the conceptual genetic interaction network.

C. Efficiency Evaluation on Real Networks

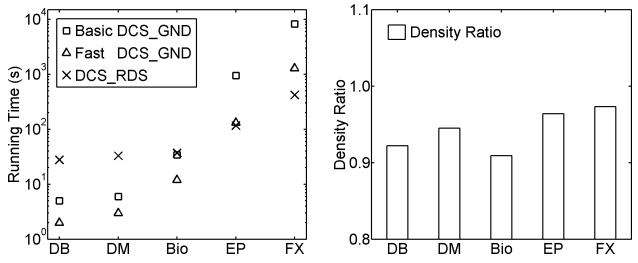
We first evaluate the efficiency of the proposed DCS_RDS and DCS_GND algorithms using both real and synthetic networks.

The DCS_RDS algorithm has three major components: removing low degree nodes (RLDN) in the conceptual network, finding the densest subgraph in the remaining graph by parametric maximum flow (PMF), and refining the densest subgraph (RDS) to make it connected in the physical network.



(a) Pruning effect (b) Running time

Fig. 16. Pruning effect and running time of the DCS_RDS algorithm



(a) Running time (b) Density ratio

Fig. 17. (a) Running time of DCS_RDS, basic and fast DCS_GND; (b) Density ratio of the subgraphs identified by DCS_RDS and basic DCS_GND

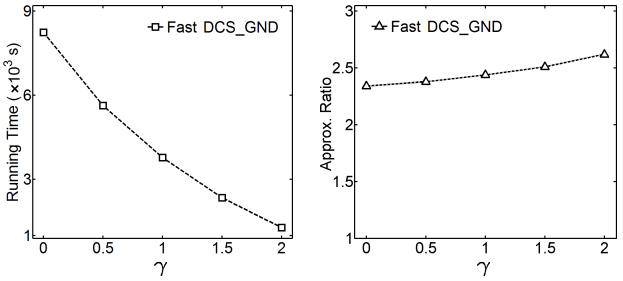
We first evaluate the pruning effect of the RLDN step. Figure 16(a) shows the number of nodes in the original graph and the number of nodes remained after pruning. It can be seen that the RLDN step can reduce the number of nodes by 2 to 4 orders of magnitude. Moreover, the pruning effect becomes larger for larger graphs. This indicates that the RLDN step is more effective when graph size increases. The effect of the optimality preserved pruning (OPP) approach discussed in Section IV is also shown in this figure. We can see that the OPP step can prune 40% to 60% nodes in the 5 real graphs. Since the real graphs are scale-free, there are many leaf nodes in the physical networks and the OPP step has large pruning ratio.

Figure 16(b) shows the running time of each step in DCS_RDS. We also run the parametric maximum flow method on the original graph (PMF-Ori) to see the performance improvement of our method. From the results, we can see that the RLDN and RDS steps run efficiently. The most time consuming part is to use parametric maximum flow to find the densest subgraph. Because of the pruning effect of RLDN, finding the densest subgraph after the RLDN step is about 2 orders of magnitude faster than directly finding it in the original graph.

Figure 17(a) shows the running time of the basic and fast DCS_GND methods. We can see that the fast DCS_GND method runs about 1 order of magnitude faster than the basic method, even though they have the same theoretical complexity. This demonstrates the effectiveness of simultaneously deleting independent non-articulation nodes. The running time of DCS_RDS is also shown in the figure for comparison. We can observe that DCS_GND runs faster on smaller graphs and DCS_RDS runs faster on larger graphs.

TABLE IV
APPROXIMATION RATIOS ON REAL NETWORKS

Dataset	DB	DM	Bio	EP	FX
DCS_RDS	1.48	1.42	1.94	1.23	2.25
Basic DCS_GND	1.53	1.44	2.11	1.26	2.34
Fast DCS_GND	2.21	2.10	2.35	1.87	2.62



(a) Running time (b) Approx. ratio

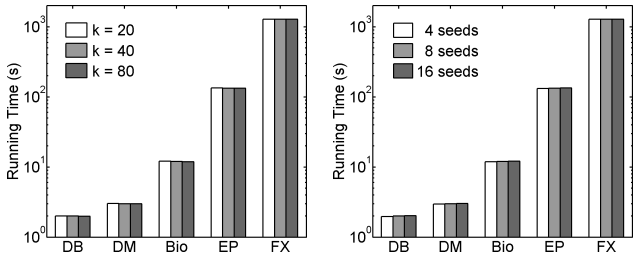
Fig. 18. Effects of γ in fast DCS_GND

The reason is that when the graph becomes larger, the depth first search procedure in DCS_GND will take longer time. On the other hand, more nodes will be removed by DCS_RDS for larger graphs as demonstrated in Figure 16(a).

Figure 17(b) shows the ratio of the density of the subgraphs identified by DCS_RDS and DCS_GND. It can be seen that the densities of the subgraphs identified by these two methods are very similar. The DCS_GND method always results slightly larger density value. The reason is that the densest subgraph in the conceptual network may not have large overlap with the DCS. The exact DCS solution may contain other dense components instead of the densest subgraph because of the connectivity constraint in the physical network.

Table IV shows the estimated approximation ratio of the proposed methods on different datasets. In the fast DCS_GND method, we set $\gamma = 2.0$. From the table, we can see that the approximation ratio of DCS_RDS is tighter than that of DCS_GND. The reason is that DCS_RDS uses the exact densest subgraph in its first step. We can also observe that the approximation ratio of the basic DCS_GND method is always smaller than that of the fast DCS_GND method. This is because the fast DCS_GND method is greedier, which deletes a set of low degree nodes in each iteration. The basic DCS_GND method only deletes the node with the minimum degree. From the table, we can also see that the approximation ratio of both methods is around 2, which is the theoretical approximation ratio of the greedy node deletion algorithm for finding the densest subgraph in a single graph.

Figure 18(a) shows the running time of the fast DCS_GND method on the Flixster dataset when varying γ. When $\gamma = 0$, it degrades to the basic DCS_GND method. When increasing γ, the fast DCS_GND method will delete more nodes in each iteration. Thus the running time decreases. Figure 18(b) shows the approximation ratio when varying γ. We can see that the approximation ratio slightly increases when increasing γ. This indicates that the approximation ratio of the fast DCS_GND method is not very sensitive to γ.

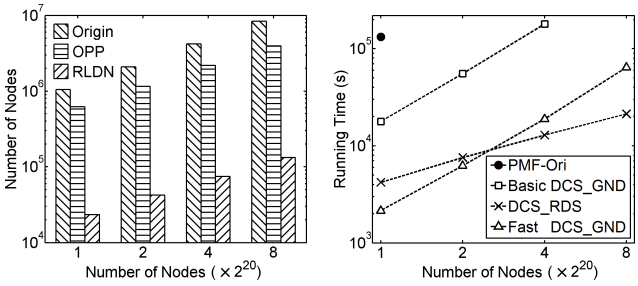


(a) Varying the size k (b) Varying the number of seeds

Fig. 19. Running time of fast DCS_GND

TABLE V
STATISTICS OF SYNTHETIC DUAL NETWORKS

#nodes	1×2^{20}	2×2^{20}	4×2^{20}	8×2^{20}
#edges in G_a, G_b	1×10^7	2×10^7	4×10^7	8×10^7



(a) Pruning ratio (b) Running time

Fig. 20. Results on synthetic dual networks

Figure 19(a) and Figure 19(b) show the running time of the fast DCS_GND method when varying the output size k in DCS_k problem and varying the number of seeds in DCS_*seed* problem respectively. From the results, we can see that fast DCS_GND has almost constant running time when varying k and the number of seeds. This is because the DCS_GND method keeps deleting nodes from the dual networks and is not sensitive to k and the number of seeds.

D. Efficiency Evaluation on Large Synthetic Networks

To further evaluate the scalability of the proposed methods, we generate a series of synthetic dual networks. Both the physical and conceptual networks are scale-free graphs based on the R-MAT model [37]. We use the graph generator from *http://www.cse.psu.edu/∼madduri/software/GTgraph/*. The statistics of the generated graphs are shown in Table V.

Figure 20(a) shows the pruning ratio of the OPP and RLDN steps in the DCS_RDS method. The OPP step can prune about 50% nodes, while the RLDN step can further reduce the number of nodes by 1∼2 orders of magnitude.

Figure 20(b) shows the running time of the DCS_RDS and DCS_GND methods. DCS_RDS has slower increasing rate. The reason is that the RLDN step has larger pruning ratio on larger graphs. The fast DCS_GND method runs about 1 order of magnitude faster than the basic DCS_GND method. This figure also shows the running time of the PMF method on the original conceptual network. PMF cannot be applied to large networks because of its long running time.

TABLE VI
APPROXIMATION RATIOS ON SYNTHETIC NETWORKS

#nodes	1×2^{20}	2×2^{20}	4×2^{20}	8×2^{20}
DCS_RDS	1.53	1.48	1.44	1.41
Basic DCS_GND	1.58	1.54	1.51	–
Fast DCS_GND	1.72	1.69	1.67	1.63

Table VI shows the approximation ratio of the DCS_RDS and DCS_GND methods. The approximation ratio becomes tighter when the size of the graph increases.

VIII. CONCLUSION

Dual networks exist in many real-life applications, where the physical and conceptual networks encode complementary information. In this paper, we study the problem of finding the densest connected subgraph in dual networks. A dense subgraph in the conceptual network that is also connected in the physical network can unravel interesting patterns that are invisible to the existing methods. We formulate the DCS problem and prove it is NP-hard. To find the DCS, we first introduce an effective optimality pruning strategy to remove the nodes that are not in the optimal solution. Then, we develop two efficient greedy algorithms to find the DCS. Extensive experimental results on real and synthetic datasets demonstrate the interestingness of the identified patterns and the efficiency of the proposed algorithms.

ACKNOWLEDGEMENT

This work was partially supported by the National Science Foundation grants IIS-1218036, IIS-1162374, IIS-0953950, the National Basic Research Program of China (No. 2014CB340401), the NIH grant R01 HG003054, the NIH/NIGMS grant R01 GM103309, and the OSC (Ohio Supercomputer Center) grant PGS0218.

REFERENCES

[1] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang, "Dense subgraphs with restrictions and applications to gene annotation graphs," in *RECOMB*, 2010, pp. 456–472.

[2] J. Chen and Y. Saad, "Dense subgraph extraction with application to community detection," *TKDE*, vol. 24, no. 7, pp. 1216–1230, 2012.

[3] V. E. Lee, N. Ruan, R. Jin, and C. Aggarwal, "A survey of algorithms for dense subgraph discovery," in *Managing and Mining Graph Data*. Springer, 2010, pp. 303–336.

[4] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan, "A fast parametric maximum flow algorithm and applications," *SIAM J. Comput.*, vol. 18, no. 1, pp. 30–55, 1989.

[5] S. Prabhu and I. Pe'er, "Ultrafast genome-wide scan for SNP-SNP interactions in common complex disease," *Genome Research*, vol. 22, no. 11, pp. 2230–2240, 2012.

[6] Y. V. Sun and S. L. Kardia, "Identification of epistatic effects using a protein-protein interaction database," *Human Molecular Genetics*, vol. 19, no. 22, pp. 4345–4352, 2010.

[7] I. Ulitsky and R. Shamir, "Pathway redundancy and protein essentiality revealed in the Saccharomyces cerevisiae interaction networks," *Mol. Syst. Biol.*, vol. 3, p. 104, 2007.

[8] R. Kelley and T. Ideker, "Systematic interpretation of genetic interactions using protein networks," *Nature Biotechnology*, vol. 23, no. 5, pp. 561–566, 2005.

[9] J. Pei, D. Jiang, and A. Zhang, "On mining cross-graph quasi-cliques," in *KDD*, 2005, pp. 228–238.

[10] H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou, "Mining coherent dense subgraphs across massive biological networks for functional discovery," *Bioinformatics*, vol. 21, no. suppl 1, pp. i213–i221, 2005.

[11] W. Li, H. Hu, Y. Huang, H. Li, M. R. Mehan, J. Nunez-Iglesias, M. Xu, X. Yan, and X. J. Zhou, "Pattern mining across many massive biological networks," in *Functional Coherence of Molecular Networks in Bioinformatics*. Springer, 2012, pp. 137–170.

[12] Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama, "Greedily finding a dense subgraph," *Journal of Algorithms*, vol. 34, no. 2, pp. 203–221, 2000.

[13] M. Charikar, "Greedy approximation algorithms for finding dense components in a graph," in *APPROX*, 2000, pp. 139–152.

[14] G. Kortsarz and D. Peleg, "Generating sparse 2-spanners," *Journal of Algorithms*, vol. 17, no. 2, pp. 222–236, 1994.

[15] B. Bahmani, R. Kumar, and S. Vassilvitskii, "Densest subgraph in streaming and MapReduce," in *VLDB*, 2012, pp. 454–465.

[16] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan, "Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph," in *STOC*, 2010, pp. 201–210.

[17] Y. Dourisboure, F. Geraci, and M. Pellegrini, "Extraction and classification of dense communities in the web," in *WWW*, 2007, pp. 461–470.

[18] P. Bedi, H. Kaur, and S. Marwaha, "Trust based recommender system for semantic web." in *IJCAI*, 2007, pp. 2677–2682.

[19] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *WSDM*, 2011, pp. 287–296.

[20] R. M. Karp, *Reducibility among combinatorial problems*. Springer, 1972.

[21] J. I. Alvarez-Hamelin, L. Dall'Asta, A. Barrat, and A. Vespignani, "K-core decomposition: a tool for the visualization of large scale networks," *arXiv preprint cs/0504107*, 2005.

[22] V. Batagelj and M. Zaversnik, "An $O(m)$ algorithm for cores decomposition of networks," *arXiv preprint cs/0310049*, 2003.

[23] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM J. Comput.*, vol. 1, no. 2, pp. 146–160, 1972.

[24] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: extraction and mining of academic social networks," in *KDD*, 2008, pp. 990–998.

[25] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *KDD*, 2008, pp. 426–434.

[26] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *RecSys*, 2010, pp. 135–142.

[27] P. Massa and P. Avesani, "Trust-aware recommender systems," in *RecSys*, 2007, pp. 17–24.

[28] P. R. Burton, D. G. Clayton, L. R. Cardon, N. Craddock *et al.*, "Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls," *Nature*, vol. 447, no. 7145, pp. 661–678, 2007.

[29] T. P. Slavin, T. Feng, A. Schnell, X. Zhu, and R. C. Elston, "Two-marker association tests yield new disease associations for coronary artery disease and hypertension," *Human Genetics*, vol. 130, no. 6, pp. 725–733, 2011.

[30] G. M. McMahon, C. M. O'Seaghdha, S.-J. Hwang, J. B. Meigs, and C. S. Fox, "The association of a single-nucleotide polymorphism in CUBN and the risk of albuminuria and cardiovascular disease," *Nephrology Dialysis Transplantation*, p. gft386, 2013.

[31] Y. Wang, J. R. O'Connell, P. F. McArdle *et al.*, "Whole-genome association study identifies STK39 as a hypertension susceptibility gene," *PNAS*, vol. 106, no. 1, pp. 226–231, 2009.

[32] P. Yue, E. Melamud, and J. Moult, "SNPs3D: candidate gene and SNP selection for association studies," *BMC Bioinformatics*, vol. 7, no. 1, p. 166, 2006.

[33] F. Luo, Y. Wang, X. Wang, K. Sun, X. Zhou, and R. Hui, "A functional variant of NEDD4L is associated with hypertension, antihypertensive response, and orthostatic hypotension," *Hypertension*, vol. 54, no. 4, pp. 796–801, 2009.

[34] K. Wang, M. Li, and H. Hakonarson, "Analysing biological pathways in genome-wide association studies," *Nat. Rev. Genet.*, vol. 11, no. 12, pp. 843–854, 2010.

[35] M.-X. Li, J. S. Kwan, and P. C. Sham, "HYST: A hybrid set-based test for genome-wide association studies, with application to protein-protein interaction-based association analysis," *AJHG*, vol. 91, no. 3, pp. 478–488, 2012.

[36] T. Ideker, O. Ozier, B. Schwikowski, and A. F. Siegel, "Discovering regulatory and signalling circuits in molecular interaction networks," *Bioinformatics*, vol. 18, no. suppl 1, pp. S233– S240, 2002.

[37] D. Chakrabarti, Y. Zhan, and C. Faloutsos, "R-MAT: A recursive model for graph mining." in *SDM*, 2004, pp. 442–446.