FacetsViewer: A Tool for Multi-faceted Decomposition of Complex Networks

Boon-Siew Seah, Sourav S. Bhowmick,
Huey Eng Chua, and Mengxuan Chen
School of Computer Science & Engg.
Nanyang Technological University
Singapore
Email: {seah0097|assourav|hechua|c110057}@ntu.edu.sg

C. Forbes Dewey, Jr.
Biological Engineering Department
Massachusetts Institute of Technology
Cambridge, MA, USA
Email: cfdewey@mit.edu

***Abstract*—The availability of large-scale network data has given rise to the opportunity to investigate higher level organization of these networks using graph theoretic analysis. In this paper, we demonstrate a novel network decomposition tool called *FacetsViewer* in order to make sense of the deluge of network data. In contrast to traditional graph clustering techniques, it finds not just a single decomposition of the network, but a *multi-faceted atlas* of semantically meaningful decompositions that portray alternative perspectives of the landscape of the underlying network. Each *facet* in the atlas represents a distinct interpretation of how the network can be meaningfully decomposed and organized. To this end, *FacetsViewer* maximizes interpretative value of the atlas by optimizing *inter-facet semantic* and *structural orthogonality*. Specifically, we demonstrate various features of *FacetsViewer* and its superior ability to generate and visualize multi-faceted atlas of complex networks.**

***Index Terms*—Attributed network, multi-faceted decompositions, semantic and structural orthogonality, visualization.**

I. Introduction

The increasing availability of massive amount of network data presents us the opportunity to comprehend high level organization of these networks. However, it is extremely difficult to extract such information by simply visualizing a network in a visual interface as it looks like a giant "hairball". Consequently, network (or graph) clustering methods that decompose a networks into their functional or topological constituents have gained increasing attention since the last decade. For example, decomposing a protein-protein interaction (PPI) network into *functional* or *topological modules* is often the key step to comprehend the high-level relationships that underlie the interaction data.

In general, graph clustering algorithms typically discover regions of dense connectivity. For instance, in a PPI network such dense regions facilitate discovery of protein complexes or functionally coherent processes. Similarly, in a social network such algorithms can facilitate detection of communities. Unfortunately, these clustering techniques output *only a single optimal decomposition* of the underlying network. Consequently, a network can only be decomposed and viewed from a single perspective, whereas in reality there are often multiple different perspectives (decompositions) associated with the high-level organization of the underlying network, all of which are distinct and equally valid. We refer to each of these decompositions as a *facet* because they visualize the organization of a network from a unique view, providing a distinct interpretation of the organization of the underlying network. Formally, a *facet* (a.k.a *decomposition* or *view*) of an undirected network $G = (V, E)$, denoted by F, is a set of *functional modules* $\{C_1, \ldots, C_m\}$ (possibly overlapping) representing a specific semantic concept. A *functional module* $C_i = (V_c^i, E_c^i)$ is a subnetwork of G such that $V_c^i \subset V$ and E_c^i is the set of edges induced by V_c^i from G. For example, consider the human PPI network. A typical decomposition based on a classical graph clustering technique identifies dense regions of the network, which correspond to the decomposition of protein complexes. However, this network can also be viewed from other different perspectives. For instance, it can be organized by the types of signaling pathways involved in it. Notice that the decomposition from this perspective is markedly different from the complex-based decomposition. Furthermore, different proteins in the network may undergo various modifications (*e.g.,* acetylation, phosphorylation, and ubiquitination). Hence, yet another way to decompose the network is by their modification effects.

At first glance, it may seem that we can tune the clustering parameters of existing graph clustering techniques in order to generate multiple facets. Unfortunately, such tuning only generates an exponential number of *slightly perturbed* decompositions with incremental differences [4]. In other words, such strategy does not generate *semantically* unique decompositions. In contrast, it is imperative to ensure that the decompositions or facets are *distinctive*, *i.e.,* they are maximally different from each other. This is because every facet should provide a fresh and informative perspective to the organization of the network, rather than providing just incremental differences w.r.t other facets.

In this demonstration, we present a novel tool called *FacetsViewer* that addresses the aforementioned limitations of classical graph clustering techniques. Specifically, it enables us to discover and visualize an *atlas* (*i.e.,* a set of facets $\{F_1, F_2, \ldots, F_n\}$ that represents distinctive semantic landscapes of G) from a network by leveraging its topology and attributes associated with its nodes, portraying alternative views of the organizational landscape of the network. Each

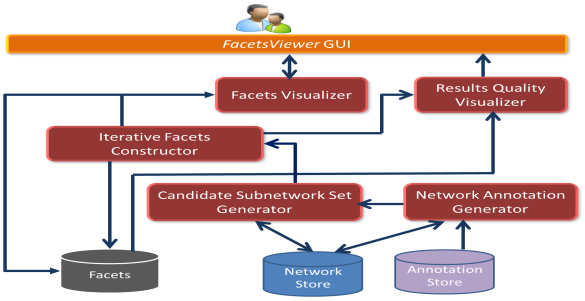

Fig. 1: System architecture of *FacetsViewer*.

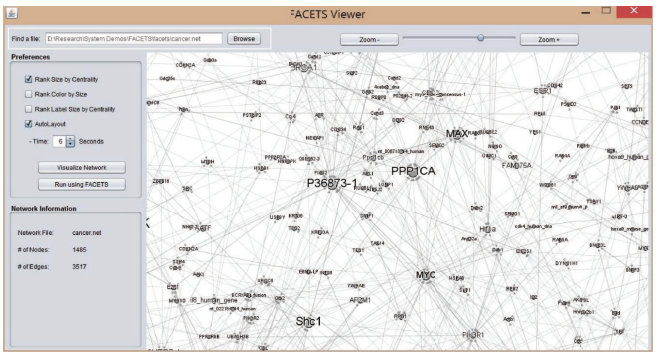

Fig. 2: GUI of *FacetsViewer*.

facet represents a distinct interpretation of how the network can be decomposed and organized. Since a key objective is to obtain n unique facets that are informative and orthogonal[1], *FacetsViewer* maximizes interpretative value of the atlas by optimizing *inter-facet semantic* and *structural orthogonality*. To elaborate further, consider a PPI network G. It can be decomposed into protein complexes based on its topological structure. On the other hand, if we consider regulatory processes as a functional (or semantic) concept, then G can be decomposed into signaling and regulation pathways, an entirely different decomposition. Furthermore, these facets must be topologically distinctive and semantically apart from each other.

II. System Architecture

Figure 1 depicts the system architecture of *FacetsViewer*, which consists of the following modules.

The GUI module. Figure 2 shows the screenshot of the *FacetsViewer* interface. A user may load an attributed network file through the top panel. The top-left panel provides a means to visualize the network by setting various visualization parameters. When the `Visualize Network` button is clicked, the right panel displays the input network based on these parameters. For example, Figure 2 shows the human cancer PPI network where the size of a node is proportional to its degree centrality. The slider in the top panel enables us to provide a zoomable view of the network. When a user clicks on the `Run with FACETS` button, the left panel is replaced by the corresponding panel shown in Figure 3. Specifically, a user may select the *ontology graph* and the number of facets to

[1]We use the term orthogonal to describe the idea of distinctive clusters, rather than its precise mathematical meaning.

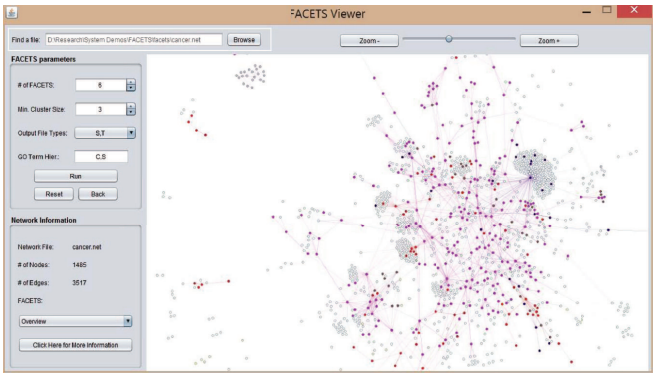

Fig. 3: Partial view of an atlas (Best viewed in color).

be discovered through the top-left panel, which are exploited by the facets generation modules. Note that in our demo we choose Gene Ontology (GO) as a representative ontology graph since we use PPI networks to demonstrate *FacetsViewer*. When the `Run` button is clicked, the right panel in Figure 3 displays the facets membership of different nodes using different colors. Note that the bottom-left panel enables us to select a specific facet for visualization using the drop-down selection box. The `Click here for more information` button displays the result quality of the multi-faceted decomposition process for the input network.

The Network Annotation Generator module. Given an undirected network $G = (V, E)$ and an ontology graph D, this module annotates the nodes of G with ontology information from D. For example, for PPI networks we utilize GO annotations associated with proteins to annotate the nodes. Specifically, given a GO directed acyclic graph $D = (V_{go}, E_{go})$, the ordered set $\Delta = \langle \Delta_1, \Delta_2, \ldots, \Delta_d \rangle$ is a topological sort of D, where Δ_i represents a single GO term. Each vertex $v \in V$ is associated with a d-dimensional *term association vector* $\Delta_v \in \{0,1\}^d$, such that $\Delta_v = \langle \Delta_1^v, \Delta_2^v, \ldots, \Delta_d^v \rangle$, $\Delta_i^v \in \{0,1\}$ where $\Delta_i^v = 1$ if and only if the term $\Delta_i \in D$ or its descendants are associated with protein v, and $\Delta_i^v = 0$ if otherwise. Note that Δ_v is an indicator vector of GO terms that are associated with v. Figure 4(a) shows a toy PPI network where the nodes are annotated with GO terms.

The Candidate Subnetwork Set Generator module. Given the annotated network $G = (V, E)$, this module generates a set of *candidate subnetworks* from G. A *facet candidate subnetwork set* $B_i = \{G_1, G_2, \ldots, G_m\}$ is a set of connected subnetworks of G such that for every $G_k \in B_i$, there is a shared ontology (*e.g.,* GO) term Δ_i within every $v \in V_k$. That is, Δ_i represents the common semantic concept (*i.e.,* function) of the candidate subnetwork. A *semantic bundle* $\omega_i = \{\Delta_1, \Delta_2, \ldots \Delta_m\}$ is the set of shared ontology annotations of B_i, *i.e.*, $\omega_i = \bigcup_{G_k \in B_i} \Delta_{G_k}$. For example, suppose B_1 is a facet candidate subnetwork set with $\omega_1 = \{\Delta_1, \Delta_2\}$, where Δ_1 represents the `Swr1 complex` GO term and Δ_2 is the `Histone` term. Hence, a subgraph of an attributed PPI network is a valid member of B_1 if every node in that subgraph is annotated with `Swr1 complex` term.

This module first creates an initial set of decompositions by performing graph clustering on G to obtain an initial set of modules[2]. Each module is then randomly associated with a facet, randomly distributing the modules over an initial set of facets. Following that, it constructs the candidate subnetworks from G that satisfy *ω_i-restricted decomposition constraint*. An *ω-restricted decomposition* is a decomposition of G into F_i such that F_i satisfies the following criteria. First, every module $C_j \in F_i$ should be *semantically bounded* by ω_i. Let $D_{C_j} = \{\Delta_1, \Delta_2, \ldots, \Delta_m\}$ be the set of shared terms in C_j. Then, the *semantic boundedness* of module C_j by ω_i is given by $r(C_j, \omega_i) = D_{C_j} \cap \omega_i$. A cluster C_j is bounded by ω_i if $r(C_j, \omega_i) \neq \emptyset$. Second, a facet F_i decomposes G by maximizing a *clustering objective function* $o(F_i)$ while satisfying the above criterion[3]. Specifically, in *FacetsViewer* every module $C_j \in F_i$ has to be structurally dense and/or semantically coherent (*i.e.,* every node in module shares a common semantic term), the coverage of F_i has to be high, and the amount of overlap between modules should be low.

Observe that an ω_i-restricted decomposition of a facet draws from a restricted search space of subnetworks in G whose vertices share at least a term within ω_i. This search space is modeled by B_i, where any valid $C_j \in F_i$ must belong to B_i. That is, for any subnetwork to be considered as a module, it must first be sharing a term in ω_i. Even if a subnetwork is dense, it must yield to sparser subnetwork candidates if it is not enriched with terms within ω_i.

Since exhaustive generation of candidate subnetworks is prohibitively expensive, we take the following steps to generate candidates for a facet F_i. For every ontology (GO) term $\Delta \in \omega_i$, we obtain the induced subnetwork in G whose nodes are annotated with Δ or its descendants. The subnetwork is then decomposed into connected components, each forming a candidate subnetwork G_j. Note that candidates formed this way can vary greatly in resolution of the annotation that its nodes share and can be highly overlapping. Fig. 4(b) and (c) show an example of the candidate subnetworks generation and initialization of facets for the toy PPI network in Fig. 4(a).

The Iterative Facets Constructor module. The goal of this module is to leverage the candidate subnetworks to simultaneously construct the atlas $A = \{F_1, \ldots, F_n\}$ of G and the *semantic partition* $\Omega = \{\omega_1, \ldots, \omega_n\}$ (a set of semantic bundles that form a partition of all ontology terms, *e.g.,* $V_{go} = \bigcup_{\omega_i \in \Omega}$), s.t the following objective function is maximized:

$$\max_{A,\Omega} \quad \lambda t(\Omega, A) + (1-\lambda)|A|^{-1} \sum_{F_i \in A} o(F_i)$$
$$\text{subject to} \quad C_s \in B_i \forall C_s \in F_i, 1 \leq i \leq n$$

The right half of the terms captures the cost function of decomposing G into A; the left half, decomposing D into Ω. The parameter λ controls the weightage between the two terms. Here $t(\Omega, A)$ is a linear combination of *inter-facet semantic* and *structural orthogonality* to ensure that every facet in the atlas A is semantically and structurally distinct modules within a facet, and as whole, structurally and semantically distinct from modules within another facet. Details related to the computation of $t(\Omega, A)$ for a PPI network is given in [8]. Observe that it is necessary to optimize these criteria simultaneously over the space of A and Ω. Otherwise, one may end up with a poor objective score. For instance, if $t(\Omega, A)$ is high (meaning highly orthogonal partitioning), but Ω is improperly partitioned such that one ends up with ω_i that allow only poor decompositions, then the $o(F_i)$ score would be very low. The reader may refer to [8] for detailed description of the optimization algorithm. Here, we briefly describe the key idea.

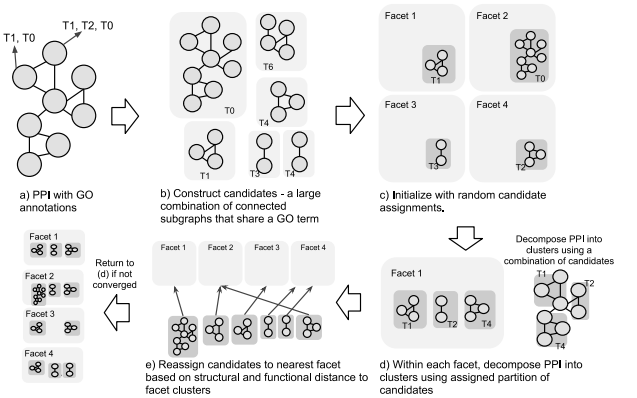

Fig. 4: Illustration of facets generation.

The optimization is performed in rounds. At each round, it updates A and Ω in two sequential steps. First, it assumes that A is a constant and update Ω to increase $t(\Omega, A)$. Specifically, for every candidate subnetwork $G_j \in B_i, 1 \leq i \leq n$, the algorithm determines its *closest* centroid by considering G_j's average *semantic* and *structural distance* to modules within a facet. Following that, G_j is reassigned to nearest B_k (superset of F_k) and Ω is updated based on where every $\Delta_j^C \in V_{go}$ is assigned to. Second, it updates A to maximize the objective function while fixing Ω. To support ω_i-restricted decomposition of every $F_i \in A$, it scores candidate subnetworks iteratively based on a profit maximization model and greedily selects the best scoring candidate as member in F_i. Figures 4(d)-(e) show the running example of these iterative steps.

The Facets Visualizer module. This module takes as input the facets generated by the preceding module and displays them visually on the network. It provides two mode of visualization, namely *Atlas View* and *Facet View*. In the *Atlas View* mode, *all* facets (*i.e.,* atlas) are shown on the network by assigning different color codes for nodes in different facets. Note that nodes that belong to multiple facets are colored black. *Atlas View* is enabled when a user selects the `Overview` item from the drop-down selection box in the bottom-left panel of the GUI. Figure 3 shows an example of the *Atlas View* in the human cancer network. On the other hand,

[2]To this end, in our implementation we use the FUSE [7] algorithm.

[3]$o(F_i)$ is determined by the specific graph clustering algorithm (FUSE [7] for our implementation) that is adapted for creating a facet.

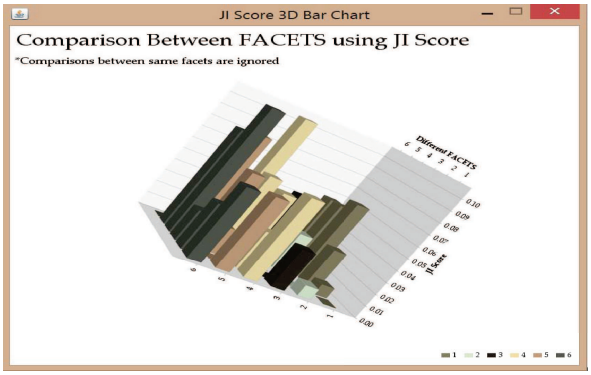

Fig. 5: Result quality visualization.

the *Facet View* allows us to view a *specific* facet by selecting it from the drop-down selection box in Figure 3.

The Result Quality Visualizer module. This module provides a real-time graphical view of the quality of facets generated for a specific network. Specifically, it measures the inter-facet decomposition similarity using *Jaccard index (JI) score*. Given two decompositions (or facets) F_1 and F_2, the *Jaccard index* (JI) is defined as $J(F_1, F_2) = \frac{A}{A+B+C}$, where A is the number of node pairs that is co-clustered in both F_1 and F_2, B is the number of node pairs co-clustered in F_1 but not F_2, and C is the number of node pairs co-clustered in F_2 but not F_1. For example, Figure 5 shows the 3D view of JI scores of the six facets discovered in the human cancer network. Observe that the low scores between facets show that they are decomposed distinctively. This reflects significant organizational differences between modules of signaling pathways and protein complexes.

This module also displays the *coverage* of a facet and the *extent* of coverage overlap between the facets (we do not show the visualization here due to space constraints). Specifically, the coverage of a facet is $Cvg(F_k) = |\bigcup_{V_c \in F_k} V_c|$. The extent of coverage overlap between F_i and F_j is $Ext(F_i, F_j) = \frac{|V_i \cap V_j|}{|V_i|}$, where $V_i = \bigcup_{V_c \in F_i} V_c$ and $V_j = \bigcup_{V_c \in F_j} V_c$. The extent of overlap between facets reaches up to 0.316 for the cancer PPI network. Hence, the overlap is not insignificant, implying that the facets are not simply partitions of G.

III. Related Systems & Novelty

Traditional network clustering techniques [9] focus on generating only a single optimal decomposition. There have also been efforts in multi-view clustering [3], [5], [6] and meta-clustering [2]. All these approaches, however, assume data points in the *vector space* that allow the notion of metric distances in a Euclidean geometry. On the other hand, *FacetsViewer* demands a multi-view clustering methodology on attributed graphs, which requires a graph clustering paradigm on both structure and annotation. To the best of our knowledge, multi-view clustering paradigm in decomposing networks from multiple, distinct perspectives has not been demonstrated in any major conference venue.

Ensemble clustering methods generate an ensemble of near-optimal decompositions [1], [4]. These near-optimal decompositions, however, have no notion of the orthogonality. Instead, ensemble clusterings create a large number of perturbed solutions, making them unsuitable as an atlas of semantically distinct decompositions. For instance, in [4], a small network of 32 nodes generated at least 82 permutations of clusterings.

IV. Demonstration Objectives

FacetsViewer is implemented in Scala and Java. It can be downloaded from https://sites.google.com/site/cosbyntu/softwares/facets. Our demonstration will be loaded with a few popular real PPI network datasets (*e.g.,* human, yeast) from *IntAct* (www.ebi.ac.uk/intact/) containing up to around 9,000 nodes. The key objective of the demonstration is to enable the audience to interactively experience the following modules through the GUI.

Facets generation and display. One of the key objectives of the demonstration is to enable the audience to interactively experience the *Iterative Facets Generator* and *Facet Visualizer* modules. Specifically, a user may choose a network using the top panel in the GUI. Then, she can invoke the facets generation module by clicking on the `Run Using FACETS` button. She can choose the number of facets as well as the ontology features from GO from the top-left panel of Figure 3 and interactively view the atlas and individual facets by interacting with the bottom-left panel. She may zoom in to a region of the network using the slider at the top panel to see details related to these facets. She can also modify the various parameters and ontology details through the top-left panel of Figure 3 and interactively experience its impact on the facets generation in the right panel. Through this experience, users will be able to appreciate the limitation of classical graph clustering techniques that only produce one unique decomposition of the network. Furthermore, we believe that this interaction will also trigger interesting discussions on some of the open research challenges associated with this problem such as realizing it on massive networks.

Result Quality Visualization. Users can also experience the working of the *Result Quality Visualizer* module by clicking on the `Click here for more information` button in the bottom-left panel to visualize the quality of generated facets in real time. One may modify the various parameters in the top-left panel (Figure 3) and experience its impact on the result quality.

References

[1] G. Agarwal and D. Kempe. Modularity-maximizing graph communities via mathematical programming. *Eur. Phys. J. B*, 66(3), 2008.
[2] R. Caruana, *et al.* Meta Clustering. *In ICDM*, 2006.
[3] Y. Cui *et al.* Non-Redundant Multi-View Clustering Via Orthogonalization. *In ICDM*, 2007.
[4] S. Navlakha and C. Kingsford. Exploring biological network dynamics with ensembles of graph partitions. *In Pacific Symposium on Biocomputing*, 2010.
[5] D. Niu and J.G. Dy. Multiple Non-Redundant Spectral Clustering Views. *In ICML*, 2010.
[6] Z. Qi and I. Davidson. A principled and flexible framework for finding alternative clusterings. *In SIGKDD*, 2009.
[7] Boon-Siew Seah, Sourav S Bhowmick, et al. FUSE: Towards multi-level functional summarization of protein interaction networks. *ACM BCB*, 2011.
[8] B.S. Seah, S. S. Bhowmick. C. F. Dewey Jr. FACETS: Multi-faceted functional decomposition of protein interaction networks. *Bioinformatics*, 28(20), 2012.
[9] Y. Zhou, H. Cheng, J. X. Yu. Clustering large attributed graphs: an efficient incremental approach. *In ICDM*, 2010.