JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 13, Number 7, 2006
© Mary Ann Liebert, Inc.
Pp. 1299–1322

Detecting Conserved Interaction Patterns in Biological Networks

MEHMET KOYUTÜRK,[1] YOHAN KIM,[2] SHANKAR SUBRAMANIAM,[2,3]
WOJCIECH SZPANKOWSKI,[1] and ANANTH GRAMA[1]

ABSTRACT

Molecular interaction data plays an important role in understanding biological processes at a modular level by providing a framework for understanding cellular organization, functional hierarchy, and evolutionary conservation. As the quality and quantity of network and interaction data increases rapidly, the problem of effectively analyzing this data becomes significant. Graph theoretic formalisms, commonly used for these analysis tasks, often lead to computationally hard problems due to their relation to subgraph isomorphism. This paper presents an innovative new algorithm, MULE, for detecting frequently occurring patterns and modules in biological networks. Using an innovative graph simplification technique based on ortholog contraction, which is ideally suited to biological networks, our algorithm renders these problems computationally tractable and scalable to large numbers of networks. We show, experimentally, that our algorithm can extract frequently occurring patterns in metabolic pathways and protein interaction networks from the KEGG, DIP, and BIND databases within seconds. When compared to existing approaches, our graph simplification technique can be viewed either as a pruning heuristic, or a closely related, but computationally simpler task. When used as a pruning heuristic, we show that our technique reduces effective graph sizes significantly, accelerating existing techniques by several orders of magnitude! Indeed, for most of the test cases, existing techniques could not even be applied without our pruning step. When used as a stand-alone analysis technique, MULE is shown to convey significant biological insights at near-interactive rates. The software, sample input graphs, and detailed results for comprehensive analysis of nine eukaryotic PPI networks are available at *www.cs.purdue.edu/homes/koyuturk/mule*.

Key words: graph mining, frequent subgraph discovery, evolution, modular conservation.

1. INTRODUCTION

INCREASING AVAILABILITY of experimental data relating to biological sequences, coupled with efficient tools such as BLAST and CLUSTAL have contributed to fundamental understanding of a variety of

[1] Department of Computer Sciences, Purdue University, West Lafayette, Indiana.
[2] Department of Chemistry and Biochemistry, University of California at San Diego, La Jolla, California.
[3] Department of Bioengineering, University of California at San Diego, La Jolla, California.

biological processes (Altschul et al., 1997; Thompson et al., 1994). These tools are used for discovering common subsequences and motifs, which convey functional, structural, and evolutionary information. Recent developments in molecular biology have resulted in a new generation of experimental data that bear relationships and interactions between biomolecules (Hartwell et al., 1999; Oltvai and Barabási, 2002). Biomolecular interaction data, generally referred to as biological or cellular networks, are frequently abstracted using graph models. Although vast amounts of high-quality data is becoming available, efficient analysis counterparts to BLAST and CLUSTAL are not readily available for such abstractions.

It is possible to model biological networks using various graph theoretic formalisms (Olken, 2003). As is the case with sequences, two key problems on graphs derived from biomolecular interactions correspond to aligning multiple graphs and finding frequently occurring subgraphs in a collection of graphs. Solutions to these problems provide understanding of several interesting concepts such as common motifs of molecular interactions, evolutionary relationships and differences among cellular network structures of different organisms, organization of functional modules, relationships and interactions between sequences, and patterns of gene regulation.

In this paper, we address the problem of finding frequently occurring molecular interaction patterns among different organisms, i.e., mining a collection of biological networks for frequent subgraphs. This problem, generally referred to as graph mining, is particularly challenging because it relates to the NP-hard subgraph isomorphism problem. Consequently, domain-specific abstractions are necessary in order to simplify the problem. We use here an abstraction based on contraction of nodes that correspond to orthologous biomolecules. We show that this simplifies the graph mining problem considerably, while being able to capture the underlying biological information accurately. Furthermore, we reinterpret the mining problem in the context of cross-species analysis of molecular interaction data to identify not only frequently occurring patterns of molecular interactions but also sets of organisms that share common interaction patterns. This facilitates phylogenetic analysis of modularity in cellular networks.

We devise an efficient algorithm, MULE, which is based on frequent itemset extraction to discover frequent subgraphs among these graphs taking into account the nature of molecular interaction data. Existing formulations of isomorphism based frequent subgraph extraction suffer from exponential explosion of problem size due to NP-hardness of both mining and subgraph isomorphism problems. In contrast to such extant approaches, MULE avoids the repeated solution of NP-hard subgraph isomorphism problem while preserving the biological relevance of identified patterns.

Using the proposed algorithm, we mine protein-protein interaction networks derived from DIP, BIND, and KEGG databases. We show that MULE is able to discover biologically meaningful patterns within seconds. In the results section, we discuss a selection of interesting patterns in detail. We also compare the computational efficiency of MULE with existing graph mining algorithms. As a stand-alone analysis technique, MULE conveys significant biological insights at rates several orders of magnitude faster than isomorphism-based graph mining algorithms. We also establish our graph simplification technique as a pruning heuristic, which may be used to discover contracted patterns to filter the data to be mined for isomorphic patterns. When used as a pruning heuristic, MULE reduces effective graph sizes significantly, accelerating existing techniques by several orders of magnitude.

We start our discussion by introducing common graph-theoretic models for biological networks and existing approaches for comparative analysis of these networks in the following section. We then provide an overview of the graph mining problem and existing graph mining algorithms in Section 2.3. In Section 3, we formalize the graph mining problem, present challenges, and propose solutions in the context of molecular interaction networks. We present and discuss the interaction patterns that result from mining KEGG metabolic pathways, and DIP and BIND protein interaction networks in Section 4. We also illustrate the runtime characteristics of the proposed algorithm. Finally, in Section 5, we derive conclusions and outline avenues for future research.

2. BACKGROUND

Before presenting the ortholog-contraction based mining algorithm for molecular interaction networks, we briefly introduce graph-theoretic formalisms for the analysis of biological networks and discuss existing literature on the comparative analysis of molecular interactions through these abstractions. We then discuss existing graph mining algorithms and challenges associated with graph-structured data.

2.1. *Graph-theoretic formalisms for molecular interaction networks*

In the multi-layered organization of living organisms, cellular interactions form the bridge between individual molecules (e.g., genes, mRNA, proteins and metabolites) and large-scale organization of the cell (Oltvai and Barabási, 2002). Common abstractions for cellular interactions include protein interaction networks, gene regulatory networks, metabolic pathways, and signaling pathways. Protein-protein interactions are obtained through high-throughput screening (Gavin et al., 2002; Ho et al., 2002; Ito et al., 2001; Amy et al., 2002) as well as small-scale experiments and have recently been organized into public databases (Bader et al., 2001; Xenarios et al., 2002). These interactions are conveniently modeled by using undirected graphs (Olken, 2003). Gene regulatory networks, also referred to as genetic networks, represent regulatory interactions between pairs of genes and are generally modeled as Boolean networks. Metabolic pathways characterize the process of chemical reactions that, together, perform a particular metabolic function and have also been organized into public databases recently (Goto et al., 1997; Karp and Mavrovouniotis, 1994; Krishnamurthy et al., 2003). A natural mathematical model for metabolic pathways is a directed hypergraph in which each node corresponds to a compound, and each hyperedge corresponds to a reaction (or equivalently enzyme) (Krishnamurthy et al., 2003). Much like metabolic pathways and gene regulatory networks, signaling pathways can also be modeled by directed graphs (Said et al., 2002).

2.2. *Comparative analysis of molecular interaction networks*

Graph alignment and graph mining provide various opportunities for cross-species analysis of biological networks. An interesting approach to understanding evolutionary conservation of interactions is the investigation of common topological motifs in molecular interaction networks (Berg and Lassig, 2004; Lotem et al., 2004; Wuchty et al., 2003). These studies reveal that more complex graph structures such as relatively large cliques or cycles are significantly conserved *in nature*, compared to simple motifs. Furthermore, proteins that are organized in cohesive patterns tend to be conserved to a higher degree. These results motivate the investigation of conserved interaction patterns among evolutionarily related proteins.

Recently, several algorithms have been proposed for the alignment of protein interaction networks to understand the conservation of pathways, complexes, and modules among different organisms (Kelley et al., 2003; Koyutürk et al., 2005; Pinter et al., 2005; Sharan et al., 2004). While comparing two networks that belong to two separate species, these methods generally construct alignment graphs by creating supernodes from any pair of potentially orthologous proteins and search for heavy subgraphs or pathways in these graphs. A similar approach is adapted to the alignment of networks that belong to multiple species, and is shown to provide significant insight on the conservation of a number of biological processes (Sharan et al., 2005). One major problem associated with large-scale application of these approaches in the comparative analysis of growing number of interaction networks is computational scalability. Since the number of nodes in the alignment graph is exponential with respect to the number of organisms, direct application of these approaches is infeasible, as interaction data for more organisms becomes available. In this study, we focus on a graph-mining approach for comparative analysis of biological networks. Since graph mining algorithms are designed for analyzing a large number of graphs, our approach is well suited to comprehensive analysis of a large collection of networks that belong to various species. By taking advantage of the nature of biological networks, we devise algorithms that render the graph mining problem tractable through contraction of orthologous nodes. While the focus here is on discovering any frequent subgraph that is connected, the method can be extended relatively easily to particular target topological structures such as linear chains (pathways) or dense subgraphs (complexes or modules) (Hu et al., 2005).

2.3. *Overview of graph mining algorithms*

There have been significant efforts aimed at developing efficient algorithms for mining graph structured datasets in recent years (Washio and Motoda, 2003). Given a collection of graphs in which nodes correspond to data items and edges to their underlying relations, we can define the graph mining problem as one of finding frequent isomorphic substructures, mapped to each other consistently with the labeling of nodes and edges.

2.3.1. Computational challenges in graph mining. Most graph mining algorithms in the literature are based on the well-studied association rule mining, or more generally, the frequent itemset problem (Hipp et al., 2000). This problem can be defined as follows. Given a set of items $\mathcal{S} = \{i_1, i_2, \ldots, i_n\}$ and a set of transactions $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ over $\mathcal{S}$, i.e., $T_i \subset \mathcal{S}$ for all i, find all subsets t of $\mathcal{S}$ such that $\sigma(t) = \frac{|\{T_i \in \mathcal{T} : t \subset T_i\}|}{|\mathcal{T}|} \geq \sigma^*$. Here, $\sigma(t)$ is the support of itemset t and σ^* is the prescribed threshold on support, signifying the desired frequency of patterns to be mined. Frequent itemset mining algorithms are generally based on the lattice or downward closure property of support. This property states that an itemset cannot be frequent if even one of its subsets is not frequent (Agrawal and Srikant, 1994). Taking advantage of this property, frequent itemset mining algorithms enumerate all potentially frequent itemsets by effectively pruning the search space. In terms of graph mining, downward closure translates to the fact that a subgraph is frequent only if all of its subgraphs are frequent.

Not surprisingly, most existing graph mining algorithms generalize state-of-the-art frequent itemset mining algorithms to structured data. However, this generalization poses significant challenges for the following reasons:

- *Subgraph Isomorphism*: While counting frequencies of subgraphs in the graph database, one must verify whether a given structure is a subgraph of a graph in the database (Washio and Motoda, 2003). This requires solution of the NP-complete subgraph isomorphism problem (Ullman, 1976) at all explored points of the solution space.
- *Canonical Labeling*: Frequent itemset mining algorithms dictate a lexicographic order on items and represent itemsets as ordered sets to ensure that no itemset is considered more than once. However such an ordering of nodes and/or edges in graphs is not trivial. Furthermore, computing canonical labels for graphs in order to sort them in a unique and deterministic manner is equivalent to testing isomorphism between graphs (Kuramochi and Karypis, 2001). Therefore, graph mining algorithms generally aim to minimize redundancy caused by duplicate consideration of subgraphs (Yan and Han, 2003).
- *Connectivity*: While taking advantage of the downward closure property in frequent itemset mining, candidate itemsets are generated in a bottom-up fashion by extending itemsets with addition of items one by one. In the case of graph mining, extension of subgraphs is not trivial since it is necessary to maintain connectivity of candidate subgraphs, since the target frequent patterns are desired to be connected, in general.

2.3.2. Existing graph mining algorithms. One of the earliest graph mining algorithms, Subdue (Cook and Holder, 2000), is based on recursively finding a subgraph that provides the best compression based on the Minimum Description Length (MDL) principle. At each step of the algorithm, the subgraph that provides maximal compression, hence is most frequent, is discovered via a beam search heuristic and replaced by a single node. This mining process is carried on recursively. In contrast to this greedy algorithm, other existing graph mining algorithms are aimed at discovering all frequent patterns, searching the entire space of subgraphs.

Another early graph mining algorithm, AGM (Inokuchi et al., 2000), adapts the well-known a-priori algorithm (Agrawal and Srikant, 1994) to mining vertex sets that induce frequent subgraphs in a graph database. The main feature of this algorithm is that it provides a canonical labeling for graphs based on an adjacency-matrix representation. This might be computationally infeasible for applications involving large graphs as in our case. FSG (Kuramochi and Karypis, 2001), on the other hand, provides a canonical representation based on sparse adjacency list data structure and adopts a breadth-first enumeration algorithm for discovering frequent subgraphs. Recent graph mining techniques are aimed at improving these algorithms by developing more efficient canonical representations that reduce redundancy in candidate generation along with several optimization techniques to help prune the search space more efficiently.

gSpan (Yan and Han, 2002) reduces the overhead introduced by the problems discussed in the previous section through a DFS-based canonical representation of graphs and enumerates the search space in a depth-first manner to achieve significant speed-up over earlier algorithms such as FSG. CloseGraph (Yan and Han, 2003) is an extension of gSpan designed to discover only those subgraphs that do not have a supergraph of same support to avoid redundancy in the output. FFSM (Huan et al., 2004a) improves upon gSpan by reducing redundant candidate generation through a vertical search scheme based on an algebraic graph framework. A recent algorithm, SPIN (Huan et al., 2004b), further speeds up graph mining by splitting the process into two independent tasks of mining subtrees and extending these subtrees to frequent subgraphs. This is based on the observation that major problems in graph mining are caused by the existence of cycles and a majority of these problems can be handled efficiently by avoiding cycles. GASTON (Nijssen and Kok, 2004) relies on the same idea to generate frequent substructures hierarchically by starting from paths, extending frequent paths to trees, and further extending frequent trees to graphs.

Ghazizadeh and Chawathe (2001) present an alternate approach for pruning the search space using summaries. In this method, graphs are summarized by superposing identically-labeled nodes and assigning weights to edges based on this superposition. Observing that the edges of a frequent subgraph must have weights greater than the frequency threshold ($\xi^* = \sigma^*|\mathcal{T}|$), it is possible to prune out many subgraphs immediately by simply evaluating the weights of the edges. Our approach in this paper also relies on the idea of contracting identically labeled nodes; however, our algorithm is particularly designed for biological networks, in which labeling of nodes does not necessarily induce a disjoint categorization. In this respect, appropriate labeling of nodes and subsequent contraction allows us to completely avoid the subgraph isomorphism problem, while preserving the underlying biological information. Moreover, in the case of biological networks, the database consists of several large graphs, while most of the existing graph mining algorithms are devised for either a large number of smaller graphs (Kuramochi and Karypis, 2001; Yan and Han, 2002) or a single large graph (Ghazizadeh and Chawathe, 2001).

3. METHODS

We first define the graph mining problem in the context of molecular interaction networks. We then introduce the idea of contracting orthologous nodes and discuss the validity and interpretability of the idea in the context of protein interaction networks and metabolic pathways. Finally, we explore strategies for adapting frequent itemset mining algorithms to mining frequent edge sets and develop an efficient algorithm (Koyutürk et al., 2004), MULE, for frequent subgraph discovery in biological networks along these lines.

3.1. Graph mining problem

This paper addresses the graph mining problem in the context of biological networks. The input to the problem is a set of graphs in which nodes correspond to biomolecules and edges correspond to interactions between these molecules. Over this set of graphs, we are looking for frequent subgraphs that are connected and isomorphic to each other. In the general setting for graph mining, isomorphism is defined with respect to the labeling of nodes. In the context of biological networks, labeling is based on the assessment of functional correspondence, as suggested by sequence homology or more comprehensive methods of functional annotation. For metabolic pathways, the hierarchical classification of enzymes provides a means for labeling nodes. In the context of protein interaction networks, proteins of different species are functionally associated through ortholog clustering. Without loss of generality, in the following discussion, we refer to nodes as proteins, and label these nodes based on the assignment of these proteins into ortholog groups. Assessment of functional correspondence between biomolecules is discussed in detail in the next section. In the following, we do not consider edge labels (e.g., compounds for metabolic pathways) for simplicity since it is relatively straightforward to extend typical graph mining algorithms to this case. We also assume that the graphs are directed, since some molecular interactions are directed (e.g., enzyme-enzyme interactions) and any undirected graph may be represented as a directed graph.

Definition 1. Interaction network. *Given a set of biomolecules V in one particular organism, a set of interactions E between these molecules, and a many-to-many mapping of these biomolecules into a set of ortholog groups $\mathcal{L} = \{l_1, l_2, \ldots, l_n\}$, the corresponding interaction network is modeled by a labeled graph $G = (V, E, \mathcal{L})$. Each $v \in V(G)$ is associated with a set of ortholog groups $L(v) \subseteq \mathcal{L}$. Each edge $uv \in E(G)$ represents an interaction between u and v.*

We define node labeling flexibly to allow proteins to be associated with more than one ortholog group. This is motivated by the fact that some proteins may be involved in more than one cellular process. Specifically, if domain families (Bateman et al., 2004; Heger and Holm, 2003) are used to relate proteins, multi-label nodes are necessary for handling multi-domain proteins. Furthermore, since observed interaction networks represent a superposition of dynamically organized interactions in spatial and temporal dimensions (Han et al., 2004), this model accurately captures the dynamic and complex modular organization of cellular processes.

Definition 2. Subgraph of an interaction network. *A graph S is a subgraph of interaction network G, i.e., $S \sqsubseteq G$ if there is an injective mapping $\phi : V(S) \rightarrow V(G)$ such that for all $v \in V(S)$, $L(v) \subseteq L(\phi(v))$ and for all $uv \in E(S)$, $\phi(u)\phi(v) \in E(G)$.*

A subgraph S is connected if and only if for any subset $U \subset V(S)$, $\exists\ u \in U$ and $v \in V(S) \setminus U$ such that $uv \in E(S)$ or $vu \in E(S)$. In molecular interaction networks, a connected graph may be interpreted as a set of interactions related to each other through at least one molecule. Therefore, interactions that are related to a particular cellular process are expected to form a connected subgraph. Such subgraphs may also be connected to each other as a reflection of crosstalk between different processes. For this reason, we define the graph mining problem as one of identifying all connected subgraphs that exist in at least a desired number of organisms. This allows us to understand the conservation and divergence of functional modules in different organisms and identify conserved links between different cellular processes.

Definition 3. Closed frequent subgraph discovery.

Input: *A set of interaction networks $\mathcal{G} = \{G_1 = (V_1, E_1, \mathcal{L}), G_2 = (V_2, E_2, \mathcal{L}), \ldots, G_m = (V_m, E_m, \mathcal{L})\}$, each belonging to a different organism, and a support threshold σ^*.*

Problem: *Let $H(S) = \{G_i : S \sqsubseteq G_i\}$ be the occurrence set of graph S. Find all connected subgraphs S such that $\xi(S) = |H(S)| \geq \sigma^*|\mathcal{G}|$, i.e., S is a frequent subgraph in $\mathcal{G}$ and for all $S' \sqsupset S$, $H(S) \neq H(S')$, i.e., S is closed.*

In this framework, one is interested in discovering all subgraphs that are frequent and closed. A closed subgraph is a frequent subgraph such that none of its supersets occur in the same set of organisms as itself. In other words, since the subgraphs of a pattern that occur in the same set of networks can be inferred from the larger pattern, reporting such subgraphs as frequent patterns would be redundant. Hence, by requiring the identified frequent subgraphs be closed, we ensure maximality of discovered patterns to avoid redundancy. Note that, in traditional mining algorithms, a closed subgraph is defined as a frequent subgraph such that none of its supergraphs is as frequent as itself. For mining biological networks, we use a generalized definition of a closed subgraph that takes into account the occurrence set of a subgraph rather than its cardinality. This allows us to identify conserved patterns for any subset of organisms, facilitating phylogenetic analysis of modularity in molecular interaction networks. This approach may also be viewed as a symmetric mining problem, where for any sufficiently large set of organisms, all *maximal* subgraphs that are common to the corresponding networks are of interest.

As can be inferred from the definition of a subgraph, our graph mining problem requires repeated solutions to the subgraph isomorphism problem. In typical applications of mining biological networks, it is necessary to run repeated queries interactively with different parameters until a satisfactory set of results is obtained. This is clearly not feasible in the current problem setting. It is important to note that there exist many proteins in an organism that are homologous to each other. This translates to the repetition of each label in a single interaction network. This is the underlying source of the subgraph isomorphism problem. As we shall now show, if all orthologous nodes are contracted into a single node, the underlying problem can be considerably simplified *while the underlying biological information is preserved.*

3.2. Ortholog contraction

We propose an alternate setting for graph mining based on contraction of orthologous nodes. While simplifying the graph mining problem significantly, ortholog contraction maintains not only the correctness by preserving the underlying frequent subgraphs in the graph database, but also the biological relevance and interpretability of the discovered patterns. The fact that the underlying frequent subgraphs in the database are preserved is formally shown, and is particularly important to note. There is *no loss of information* resulting from our ortholog clustering technique.

Definition 4. Ortholog-contracted graph. *Given interaction network $G = (V, E, \mathcal{L})$ the ortholog-contracted representation of G, $\Upsilon(G) = \bar{G} = (\bar{V}, \bar{E}, \mathcal{L})$ is constructed as follows. For $1 \leq i \leq |\mathcal{L}|$, there exists unique $\bar{v} \in \bar{V}$ such that $L(\bar{v}) = \{l_i\}$. For each $uv \in E$ and for all $l_i \in L(u)$, $l_j \in L(v)$, there exists $\bar{u}\bar{v} \in \bar{E}$ such that $L(\bar{u}) = \{l_i\}$ and $L(\bar{v}) = \{l_j\}$.*

A sample interaction network and its ortholog-contracted representation are shown in Figure 1. Observe that the ortholog-contracted graph of an interaction network is unique while the reverse is not necessarily true. However, all subgraphs of an interaction network are preserved in its ortholog-contracted representation, as the ortholog-contracted representations of all subgraphs of G are subgraphs of $\bar{G}$, as stated in the following theorem.

Theorem 1. Preservation of subgraphs. *Given interaction network $G = (V, E, \mathcal{L})$, let $\Upsilon(G) = \bar{G} = (\bar{V}, \bar{E}, \mathcal{L})$ be its ortholog-contracted representation. Then for any $S \sqsubseteq G$, $\Upsilon(S) \sqsubseteq \bar{G}$.*

Proof. Take any $S \sqsubseteq G$. Let $\bar{S} = \Upsilon(S)$ and ϕ be the appropriate mapping from $V(S)$ to $V(G)$. For each $v \in V(S)$ and $l_i \in L(v)$, there exists a unique $\bar{v} \in V(\bar{S})$ such that $L(\bar{v}) = \{l_i\}$. Since $L(v) \subseteq L(\phi(v))$, $l_i \in L(\phi(v))$. Therefore, there also exists a unique $\overline{\phi(v)} \in V(\bar{G})$ such that $L(\overline{\phi(v)}) = \{l_i\}$. Then, there is a unique injective mapping $\bar{\phi} : V(\bar{S}) \rightarrow V(\bar{G})$, where $\bar{\phi}(\bar{v}) = \overline{\phi(v)}$ for any $v \in V(S)$. Hence, for any $\bar{u}\bar{v} \in E(\bar{S})$ that results from $uv \in E(S)$, since $\exists\ \phi(u)\phi(v) \in E(G)$, there exists $\bar{\phi}(\bar{u})\bar{\phi}(\bar{v}) = \overline{\phi(u)}\ \overline{\phi(v)} \in E(\bar{G})$. Therefore, $\bar{S} \sqsubseteq \bar{G}$. ■

In Figure 1, the ortholog-contracted representation of the bold subgraph of G is also shown in bold in $\Upsilon(G)$.

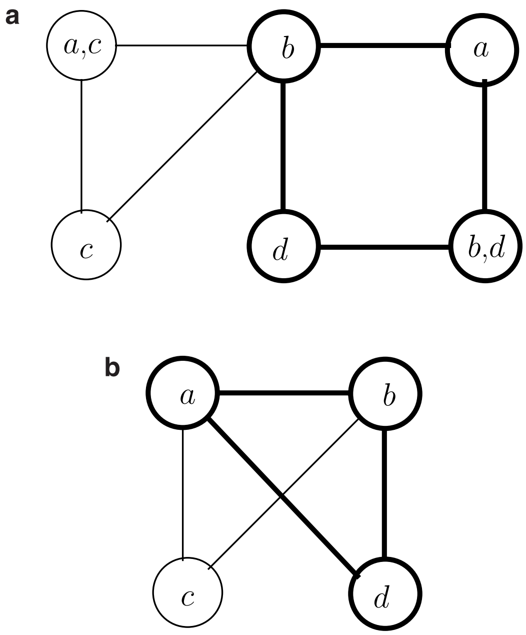

FIG. 1. Illustration of ortholog contraction. (**a**) A sample interaction network G. (**b**) Its ortholog-contracted representation $\Upsilon(G)$. The ortholog-contracted representation of the bold subgraph of G exists in $\Upsilon(G)$, also shown in bold.

Corollary 1. Preservation of frequent subgraphs. *For a set of interaction networks $\mathcal{G} = \{G_1, G_2, \ldots, G_m\}$, let $\bar{\mathcal{G}} = \{\Upsilon(G_1), \Upsilon(G_2), \ldots, \Upsilon(G_m)\}$ be the corresponding set of ortholog-contracted graphs. If S is a frequent subgraph in $\mathcal{G}$, then $\Upsilon(S)$ is a frequent subgraph in $\bar{\mathcal{G}}$.*

We can interpret this result as follows. If we mine the set of ortholog-contracted graphs instead of the original set of interaction networks, we will discover a *superset* of the frequent subgraphs of the original set. In other words, *we do not miss* any frequent patterns that exist in the dataset. Therefore, it is always possible to recover the actual frequent subgraphs from the set of frequent ortholog-contracted subgraphs using an isomorphism-based graph mining algorithm. This is significantly more efficient than running the isomorphism-based algorithm on the original dataset, since mining the ortholog-contracted graph prunes out most of the infrequent substructures, thus the resulting set is significantly smaller both in terms of graph size and number of graphs. Furthermore, the idea of ortholog-contraction does not conflict with the purpose of mining molecular interaction data; as we shall show, it is very useful by itself. We elaborate on this point in the context of metabolic pathways and protein interaction networks.

3.2.1. Ortholog contraction in protein interaction networks. Recent studies on the evolution of protein interaction networks suggest that orthologous proteins that result from recent duplications are likely to share common interactions (Wagner, 2001). In other words, conservation of interactions between orthologous proteins translates into conservation of function. Therefore, while mining protein interaction networks for common network patterns among different species, proteins in different organisms must be related to each other through orthology.

Since proteins that are evolutionarily or functionally related show significant sequence homology, a reasonable way of detecting protein families relies on sequence clustering (Enright et al., 2002; Yang and Wang, 2002). A problem with inter-species protein sequence clustering is that out-paralogs, which have no functional or evolutionary relationship since they predate the split of species, are also clustered together along with orthologs and in-paralogs (Remm et al., 2001).

Recently, ortholog families have been identified through more comprehensive *in-silico* analysis and organized into several databases, such as COG (Tatusov et al., 2003) and Homologene (Wheeler et al., 2003). There has been relevant efforts to comprehensively identify domain families as well, including PFAM (Bateman et al., 2004) and ADDA (Heger and Holm, 2003). However, in order to avoid overpopulating the contracted network, interacting domains should be considered while relating nodes in the interaction network through domain families.

Node contraction in protein interaction networks reduces interactions between proteins into those between ortholog groups (Fig. 2). A five-node portion of *S. cerevisiae* protein interaction network is shown in Figure 2a. In this figure, the common names of each protein are shown in the oval representing that protein. The nodes are labeled by their COG clusters. As a result of ortholog contraction, $3'5'$ exoribonuclease (Mtr3) and $3'5'$ phosphorolytic exoribonuclease (Ski6), which belong to the same COG family,

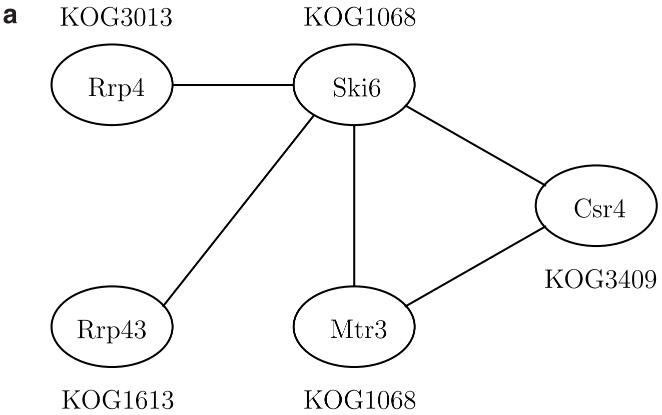

FIG. 2. Ortholog contraction in molecular interaction networks. (**a**) A five-node portion of the *S. cerevisiae* protein interaction network. Each protein is labeled by the COG cluster it belongs to. (*continues*)

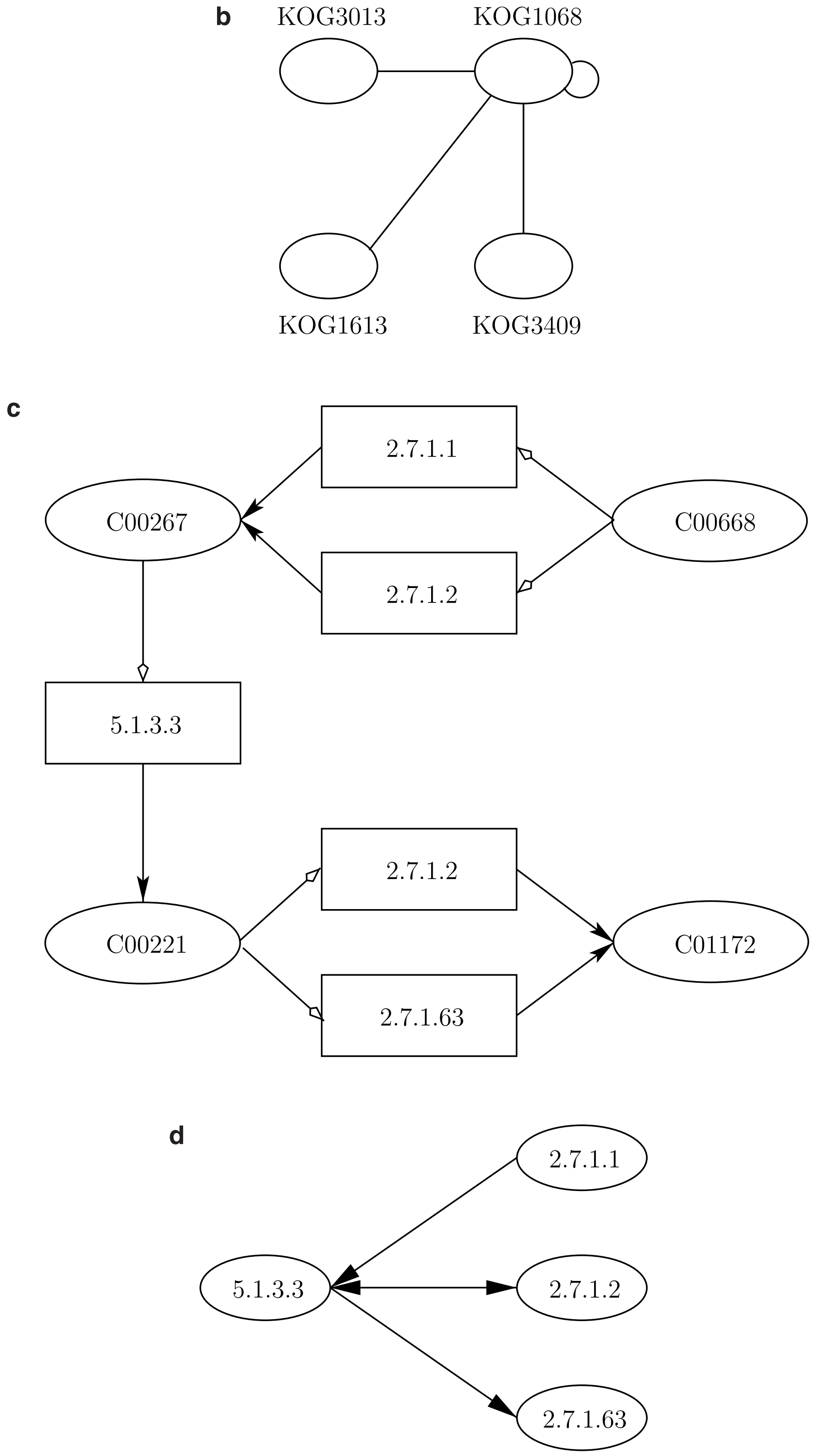

FIG. 2. (*Continued*). Ortholog contraction in molecular interaction networks. (**b**) The ortholog-contracted representation of this protein interaction network based on ortholog groups in COG. (**c**) A portion of glycolysis reference pathway in directed hypergraph representation. Compounds are shown by rectangles, enzymes are shown by ovals. For each reaction, there is an edge from each substrate to the catalyzing enzyme and one from the catalyzing enzyme to each product. (**d**) Ortholog-contracted representation of this sub-pathway.

are contracted into single node, as shown in Figure 2b. Therefore, the interaction of these proteins with Csl4 is represented as a single interaction between ortholog groups KOG1068 and KOG3409.

3.2.2. Ortholog contraction in metabolic pathways. In the directed graph model for metabolic pathways, node labels correspond to enzymes that catalyze the respective reactions. Although the biochemical properties of enzymes differ from organism to organism, enzymes are classified based on metabolic functions and protein orthologies. Currently, there exists a comprehensive enzyme nomenclature that provides hierarchical classification of enzymes based on biochemical function (Liébecq, 1992). In this enzyme nomenclature system, each enzyme is identified by its Enzyme Commission (EC) number. The numbers in the squares that represent reactions in Figure 2c are the EC numbers of the enzymes that catalyze these reactions.

An enzyme may catalyze multiple reactions in a particular pathway. Therefore, an enzyme class may be attached to more than one node in the corresponding graph model. However, since the edges in the directed graph model signify the producer-consumer relation between two enzymes, contracting nodes corresponding to the orthologous enzymes (i.e., enzymes that belong to the same class) preserves this information (Koyutürk et al., 2004). The ortholog-contracted representation of the metabolic pathway graph of Figure 2c is shown in Figure 2d. In this representation, although the node that corresponds to enzyme EC:2.7.1.2 is contracted, we do not lose the information that this enzyme not only consumes the product of EC:5.1.3.3, but also produces a compound that is consumed by the same enzyme. The only information that is hidden by this model is the fact that these two interactions between this pair of enzymes are derived from two successive reactions, which may be extracted by post-processing, as theoretically shown in the previous section.

3.3. Mining ortholog-contracted graphs for frequent edgesets

Once we contract orthologs into a single node for each graph, the frequent subgraph discovery problem is reduced to a generalized form of frequent itemset mining. We elaborate on this point in the following lemma.

Lemma 1. Equivalence of ortholog-contracted graphs to edge sets. *For ortholog contracted graph $\bar{G}$, define edge set $\tilde{E}(\bar{G}) = \{(l_i, l_j) : \exists uv \in E(\bar{G})$ such that $L(u) = \{l_i\}, L(v) = \{l_j\}\}$. If $\bar{S}$ is also an ortholog-contracted graph, then $\bar{S} \sqsubseteq \bar{G}$ if and only if $\tilde{E}(\bar{S}) \subseteq \tilde{E}(\bar{G})$.*

Proof. It is straightforward to see that if $\bar{S} \sqsubseteq \bar{G}$, then $\tilde{E}(\bar{S}) \subseteq \tilde{E}(\bar{G})$. Now assume that $\tilde{E}(\bar{S}) \subseteq \tilde{E}(\bar{G})$. For any $(l_i, l_j) \in \tilde{E}(\bar{S})$, there exist unique $u, v \in V(\bar{S})$ such that $L(u) = \{l_i\}$, $L(v) = \{l_j\}$, and $uv \in E(\bar{S})$. Furthermore, $(l_i, l_j) \in \tilde{E}(\bar{G})$. Therefore, there exist unique $u', v' \in V(\bar{G})$ such that $L(u') = \{l_i\}$, $L(v') = \{l_j\}$, and $u'v' \in E(\bar{G})$. Letting $\phi(u) = u'$ and $\phi(v) = v'$, we have $\bar{S} \sqsubseteq \bar{G}$. ■

We can generalize this lemma to conclude that an ortholog-contracted graph is uniquely determined by the set of its edges. Therefore, mining frequent subgraphs in a collection of ortholog-contracted graphs is equivalent to mining frequent edgesets in a collection of graphs that are uniquely determined by the set of their edges. Since we are interested only in connected subgraphs, we define an edgeset to be the set of label pairs that correspond to the edges of a connected graph.

Definition 5. Edgeset. *Given a set of ortholog labels $\mathcal{L} = \{l_1, l_2, \ldots, l_n\}$, an edgeset $F = \{e_1, e_2, \ldots, e_k\}$ is a set of ordered pairs $e_i = \{l_s, l_t\}$, where for any subset $F' \subset F$, there exists $e_i \in F'$, $e_j \in F \setminus F'$ such that $e_i \cap e_j \neq \emptyset$.*

Definition 6. Closed frequent edgeset discovery.

Input: *Set of ortholog contracted graphs $\bar{\mathcal{G}} = \{\bar{G}_1, \bar{G}_2, \ldots, \bar{G}_m\}$ and a support threshold σ^*.*

Problem: *For edgeset F, let $H(F) = \{\bar{G}_i : F \subseteq \tilde{E}(\bar{G}_i)\}$ be the occurrence set of F. Find all closed edgesets F that are frequent in $\bar{\mathcal{G}}$, i.e., $\xi(F) = |H(F)| \geq \sigma^*|\bar{\mathcal{G}}|$ and for all $F' \supset F$, $H(F') \neq H(F)$.*

Observe that this problem is a generalized version of the frequent itemset mining problem. Indeed, frequent itemset mining is a special case in which the underlying graph is a clique. Therefore, a simple approach to solving this problem is to remove the connectivity constraint, and find all frequent subgraphs using a frequent itemset mining algorithm. The connected components of all frequent subgraphs provide the set of all frequent connected subgraphs. However, this approach has two drawbacks. First, although it ensures that all frequent edgesets will be discovered, it does not ensure that the discovered edgesets will be closed. Second, since the number of connected subgraphs of a clique is much larger than that of a sparse graph, this relaxation will enlarge the search space significantly, degrading computational efficiency. Therefore, a specialized algorithm for this problem, which takes into account the connectivity and maximality constraints, along with the nature of data that is derived from molecular interactions is necessary.

3.3.1. Adapting itemset mining to edgeset mining. Since the frequent edgeset mining algorithm is closely related to the frequent itemset mining problem, we base our algorithm design on existing itemset mining algorithms taking into account the specific characteristics of biological networks.

As discussed in the previous section, frequent itemset mining algorithms enumerate the space of possible itemsets, exploiting the downward closure property to prune out the search space. Starting from the smallest itemsets, the occurrence of each itemset in the input transaction set is counted. Smaller frequent itemsets are extended with other frequent itemsets to generate larger itemsets that are potentially frequent. Repetitions are avoided by inducing a lexicographic ordering of items.

Two major design choices for frequent itemset mining algorithms are, the order of traversal of the enumeration tree and the method for determining the support of each itemset (Agarwal et al., 2001). It is possible to traverse the itemset tree in depth-first or breadth-first fashion. Breadth-first traversal, which generates the nodes of the tree level by level, is efficient in the sense that it eliminates the maximum number infrequent itemsets at each level. However, it requires a larger memory since it stores all nodes at each level of the tree. Therefore, breadth-first traversal becomes inefficient as the tree gets deeper. Depth-first traversal, on the other hand, expands a node immediately after its itemset is discovered to be frequent, keeping the storage requirement to a minimum, at the expense of exploring extra itemsets (Hipp et al., 2000).

There are two possible methods for computing the support of each itemset as well. One approach is the set counting method, which makes a pass over the transaction set at each node to count the number of transactions that contain the corresponding itemset. This approach is memory-efficient and well-suited to breadth-first traversal. Set intersection, on the other hand, stores the identifiers of all transactions that contain each itemset and computes the intersection of identifier sets while extending an itemset. This approach minimizes the number of passes over the transaction set at the expense of additional memory for storing the identifier sets. This method is more appropriate for depth-first traversals.

Most closed frequent itemset mining algorithms use a depth-first traversal along with set intersection, since depth-first traversal provides the opportunity of deciding whether an itemset is closed upon its expansion (Burdick et al., 2001; Gouda and Zaki, 2001). This combination is also appropriate for the closed frequent edgeset mining problem in biological networks for the following reasons:

- *Occurrence of subgraphs*. In contrast to association rule mining, in mining biological networks, the identity of organisms that contain the particular subgraph is of interest as well as its frequency. This is because, this set of organisms provides considerable information about the conservation of pathways, modules, and complexes, evolutionary relations between species, and the functional annotation of discovered interaction patterns. Therefore, for each edgeset explored by the algorithm, it is necessary to store the identifiers of organisms that contain this edgeset.
- *Graph size versus database size*. In biological applications, the size of the graphs is larger than the size of typical transactions in association rule mining. For instance, a protein interaction network generally contains thousands of edges. This is also true for the cardinality of identified patterns. On the other hand, while typical data mining applications involve millions of transactions, the number of biological networks to be mined is smaller. Therefore, in mining biological networks, the enumeration tree is wider and deeper, while the data to be processed at each enumeration node is smaller. This makes depth-first enumeration along with set intersection feasible and memory efficient.

3.3.2. Mule: An efficient algorithm for maximal frequent edgeset mining. The key difference between frequent edgeset mining and frequent itemset mining is that in the former, we are only interested in connected subgraphs. In order to generate all connected subgraphs in the database, we perform depth-first search on the graph constructed from all frequent edges. To avoid repetitions, we induce a lexicographic order on the edges and remember previously visited edges at each enumeration node. Assume, at any stage of the algorithm, that we have a frequent edgeset of k edges, denoted by F_k. We define the candidate set C_k to be the set of edges that are connected to the edges in F_k, but have not been previously visited. The set of edges previously visited by the depth-first enumeration algorithm is denoted by D_k. For any candidate edge $c \in C_k$, we extend F_k as follows:

$$F_{k+1} = F_k \cup c \qquad\qquad D_{k+1} = D_k = D_k \cup c,$$

$$N(c) = \{e \in \mathcal{F} : e \cap c \neq \emptyset\} \qquad C_{k+1} = (C_k \cup N(c)) \setminus D_k.$$

Here, $\mathcal{F}$ denotes the set of all frequent edges in the graph database.

The resulting algorithm for **M**ining **U**nique-**L**abeled **E**dgesets (Mule) is shown in Algorithm 1. This algorithm makes use of a recursive subroutine to extend frequent edgesets, which is shown in Algorithm 2.

The main procedure, MineOrthologContractedGraphs performs pre-processing by determining the set of frequent edges in the input graph set. It then generates each portion of the frequent edgeset tree rooted at each frequent edge by calling ExtendFrequentEdgeset. Upon each invocation, ExtendFrequentEdgeset tries to extend the edgeset (subgraph) by all edges in the candidate set, one by one. If the extended edgeset is frequent, then the procedure is invoked again for the extended edgeset. The algorithm stops whenever an edgeset cannot be further extended. This edgeset is then recorded, if it is not subsumed by any other recorded frequent edgeset. Upon invocation, ExtendFrequentEdgeset checks whether the current frequent tree is already subsumed by other closed frequent edgesets that have previously been discovered, if so, it stops the search process. This optimization helps prune out the search space in chunks. MFS is empty on first invocation of ExtendFrequentEdgeset, and is input to the procedure at each subsequent invocation, wherein it is extended with newly discovered frequent subgraphs.

Consider the input graph set of Figure 3a. These graphs have six edges in all, ab, ac, bd, ce, de, and ea. Figure 3b shows the frequent edgeset tree for mining subgraphs that exist in at least three of the input graphs. Procedure ExtendFrequentEdgeset is invoked for ab, ac, de, and ea, since these are the only frequent edges. The edgeset F, candidate set C, and the set H of identifiers of graphs that contain this edgeset are shown at each node of the edgeset tree. The sets of visited edges (D) label the branches of the tree, since these sets are shared by parent and children. At any instant, set D for a node is the one at its right-most branch. On first invocation, the algorithm starts with edgeset $\{ab\}$, whose candidate set is $N(ab) = \{ac, ea\}$ and extends it with edge ac since the edgeset $\{ab, ac\}$ is frequent. This set cannot be extended by the only edge in its candidate set, ea, since the edgeset $\{ab, ac, ea\}$ is a subgraph of only two input graphs. Therefore, this edgeset is recorded as a closed frequent subgraph. Note that extension of the edgeset with edge de is not considered since this edge is not connected to the edgeset under consideration. Therefore, it never gets into the candidate edge set. Furthermore, extension of the edgeset $\{ac\}$ with edge ab is not considered since this edge has already been visited. Upon termination, the algorithm reports four closed frequent subgraphs shown in boxed nodes in the figure, which are $\{ab\}$, $\{ab, ac\}$, $\{ab, ea\}$ and $\{de\}$. Note that $\{ab\}$ is reported since its occurrence set is different from its superset $\{ab, ac\}$, hence it is closed. Although edgesets $\{ac\}$ and $\{ea\}$ are also frequent, they are not reported since they are contained in other frequent edgesets with the same occurrence set.

3.4. Statistical significance

The frequency of a subgraph, which is defined as the number of networks (organisms) that contain this subgraph, provides a heuristic measure for evaluating the statistical significance of the conservation of this subgraph. This frequency threshold is an input parameter for Mule, setting a bound on this parameter to be considered significant. In order to quantify the statistical significance of identified patterns in terms of conventional p-values, we use a simple reference model, which takes the degree distribution into account. Let $X_{ij}(r)$ be the random variable indicating the existence of an interaction between ortholog groups l_i

Algorithm 1. Main procedure for mining ortholog-contracted graphs

```
procedure MineOrthologContractedGraphs ($\mathcal{G}$, $\sigma^*$)
    ▷Input $\mathcal{G}$: Set of ortholog-contracted graphs
    ▷Input $\sigma^*$: Support threshold
    ▷Output $MFS$: Set of closed frequent subgraphs
    $\xi^* \leftarrow \sigma^*|\mathcal{G}|$
    $\mathcal{E} \leftarrow \{e = \{l_s, l_t\} : \exists\ G \in \mathcal{G}\ s.t.\ u, v \in V(G),\ uv \in E(G), L(u) = l_s, L(v) = l_t\}$
    for each $e = \{l_s, l_t\} \in \mathcal{E}$ do
        $H(e) \leftarrow \{G \in \mathcal{G} : \exists\ u, v \in\ V(G)\ s.t.\ uv \in E(G),\ L(u) = l_s,\ L(v) = l_t\}$
    $\mathcal{F} \leftarrow \{e \in \mathcal{E} : |H(e)| \geq \xi^*\}$
    $MFS \leftarrow \emptyset$
    for each $e_i \in \mathcal{F}$ do
        $N(e_i) \leftarrow \{e_j \in \mathcal{F} : e_j \cap e_i \neq \emptyset\}$
        ExtendFrequentEdgeset ($\mathcal{F}$, $\xi^*$, $MFS$, $\{e_i\}$, $N(e_i)$, $\{e_1, e_2, \ldots, e_{i-1}\}$)
    return $MFS$
```

and l_j in network G_r. Assuming that all interactions in a network are independent from each other, we estimate the probability of this interaction based on the number of interactions of the two ortholog groups in the corresponding organism as $P(X_{ij}(r) = 1) = d_r(i)d_r(j)/|E(G_r)|$ (Chung et al., 2003). Here, $d_r(i)$ denotes the number of interacting partners of ortholog group i in network G_r. For a set of interactions F, let random variable $Y_F(r)$ indicate the existence of F in network G_r, i.e., $Y_F(r) = (F \subseteq E(G_r))$. Then, $P(Y_F(r)) = \Pi_{l_i l_j \in F} P(X_{ij}(r) = 1)$. Defining $Z_F = \sum_{r=1}^{m} Y_F(r)$ as the number of networks that contain F, we evaluate the significance of observing F in k networks by $P(Z_F \geq k)$, i.e., the probability of F being a subgraph of at least k networks. Assuming that the interaction networks are generated independently from each other, we directly estimate this probability for small number of networks. For larger numbers of networks, on the other hand, we estimate the z-score for the observed pattern through normal approximation.

Algorithm 2. Recursive procedure for extending a frequent edgeset

```
procedure ExtendFrequentEdgeset ($\mathcal{F}$, $\xi^*$, $MFS$, $F_k$, $C_k$, $D_k$)
    ▷Input $\mathcal{F}$: Set of frequent edges
    ▷Input $\xi^*$: Frequency threshold
    ▷Input, Output $MFS$: Set of maximal frequent edgesets
    ▷Input $F_k$: Frequent edgeset with $k$ edges
    ▷Input $C_k$: Set of candidate edges for edgeset extension
    ▷Input $D_k$: Set of already visited edges
    $R_k \leftarrow$ set of all unvisited edges reachable from $F_k$
    if $\exists F' \in MFS$ s.t. $R_k \subseteq F'$ and $H(F_k) \subseteq H(F')$ then return
    $closed \leftarrow$ true
    for each $c \in C_k$ do
        $D_{k+1} \leftarrow D_k \leftarrow D_k \cup \{c\}$
        $F_{k+1} \leftarrow F_k \cup \{c\}$
        $H(F_{k+1}) \leftarrow H(F_k) \cap H(c)$
        if $|H(F_{k+1})| \geq \xi^*$ then
            if $H(F_{k+1}) = H(F_k)$ then $closed \leftarrow$ false
            $C_{k+1} \leftarrow (C_k \cup N(c)) \setminus D_{k+1}$
            ExtendFrequentEdgeset($\mathcal{F}$, $\xi^*$, $MFS$, $F_{k+1}$, $C_{k+1}$, $D_{k+1}$)
    if $closed$ then
        if $\nexists F' \in MFS$ s. t. $F_k \subseteq F'$ and $H(F_k) \subseteq H(F')$ then $MFS \leftarrow MFS \cup F_k$
```

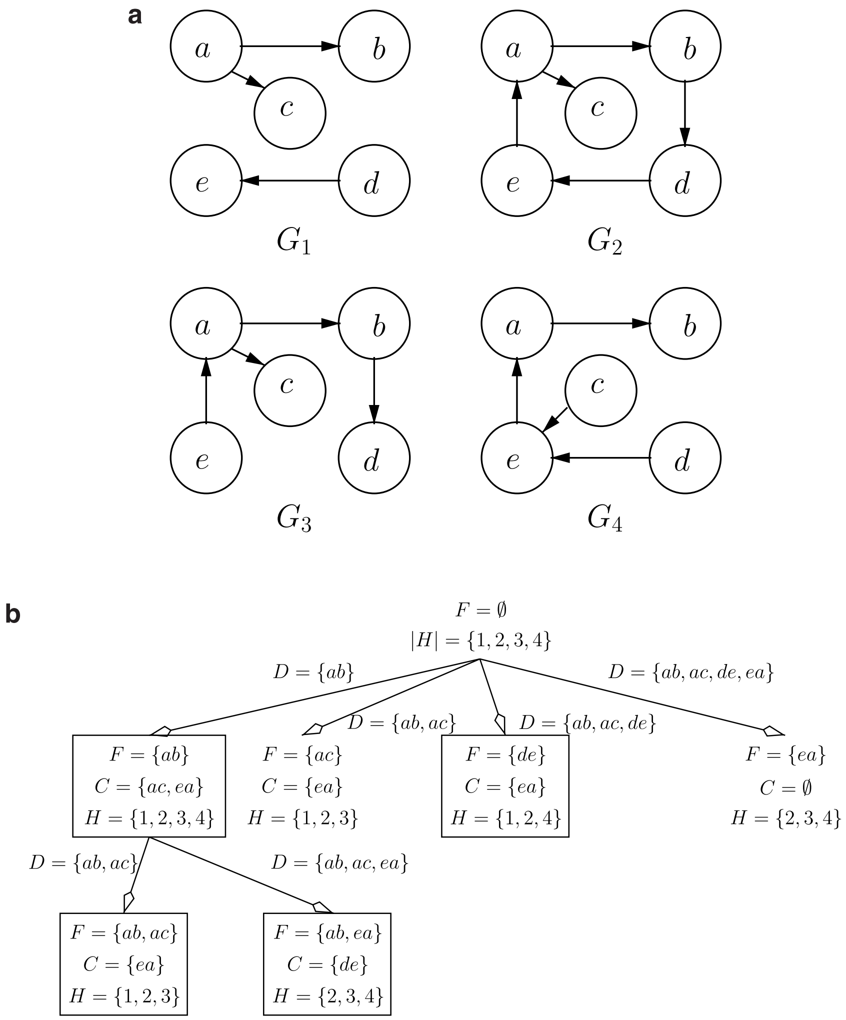

FIG. 3. Sample execution of MULE. (**a**) Input graph set. (**b**) Resulting enumeration tree of frequent edgesets.

4. RESULTS AND DISCUSSION

In this section, we first present molecular interaction patterns discovered by MULE and discuss their biological interpretation. We then illustrate the runtime efficiency of MULE, compare its execution characteristics with those of FSG and gSpan, and show that it is possible to recover actual frequent subgraphs from the contracted patterns discovered by MULE very quickly using an isomorphism-based graph mining algorithm.

4.1. Mining results

4.1.1. Frequent molecular interaction patterns in DIP protein interaction networks. In this section, we present results on mining nine eukaryotic protein interaction networks gathered from BIND (Bader et al., 2001) and DIP (Xenarios et al., 2002). In order to relate the proteins in different organisms and compute ortholog-contracted graphs, we use ortholog groups derived from COG, Homologene, and sequence clustering using BLASTCLUST. We compare each homolog group in Homologene with ortholog groups in COG. If a Homologene group shares at least one protein with a COG ortholog group, we merge the Homologene group into the corresponding COG group. We then compare each protein that is not yet assigned to an ortholog group with the existing ortholog groups using BLAST. If the protein has significant sequence similarity with at least half of the proteins in a group, then we assign the protein to that ortholog group as well. For the remaining proteins, we run BLASTCLUST and create a new ortholog group from each cluster identified by BLASTCLUST. We then compute the ortholog-contracted graphs based on these ortholog groups, considering both direct and one-hop indirect interactions. The statistics of the original PPI networks and the ortholog-contracted graphs are shown in Table 1.

When we mine the nine PPI networks for patterns that occur in at least four of the input networks, i.e., those of frequency four, we are able to identify 41 frequent connected subgraphs. The largest subgraph that is common to *H. sapiens*, *D. melanogaster*, *C. elegans*, and *S. cerevisiae* contains 18 interactions between 19 ortholog groups, which is shown in Figure 4a. These interactions are associated with zinc-finger domains (KOG1721). For any combination of three organisms among these four, we are able to obtain larger subgraphs that are related to zinc-finger proteins. For example, *H. sapiens*, *D. melanogaster*, and *C. elegans* share 115 interactions related to zinc-finger among 83 ortholog groups ($p < 5e - 206$), while *H. sapiens*, *D. melanogaster*, and *S. cerevisiae* share 81 interactions among 66 ortholog groups ($p < 3e - 152$). The star shape of this interaction network is probably due to (1) numerous cellular activities that zinc-finger proteins participate in (e.g., cell division, transcription, MAP Kinase signaling, and actin polymerization, and others) and (2) a large number of proteins with zinc-finger domains, both in higher and lower eukaryotes (about 1% of proteins in mammals [Iuchi, 2001]). Surprisingly, there is a significant degree of conservation of interactions among zinc-finger proteins and their partners across these diverse organisms. An interesting follow-up investigation would be to see how DNA binding specificities of these zinc-finger domains have evolved.

Using the same number of organisms for the threshold, a portion of a large protein complex, TFIID, involved in transcription by RNA Polymerase II is identified as a conserved subnet in *M. musculus*, *H. sapiens*, *D. melanogaster*, and *S. cerevisiae* (Auty et al., 2004). This conserved subnet is shown in Figure 4b. The mapping of these interactions on each organism are also shown in the figure, where direct and indirect interactions are shown by solid and dashed edges, respectively. In *S. cerevisiae*, this protein complex consists of one TATA-Binding Protein (TBP) and at least 14 TATA-Associated Factors (TAFs); yet in the conserved subnetwork, only four are found (Auty et al., 2004). One hypothesis explaining this observation is that the TAFs present in the conserved network have greater role in promoting transcription relative to other TAFs that are absent.

When we lower the frequency threshold to 3, MULE identifies much larger number of conserved interaction patterns, specifically 158 frequent subgraphs. Four of these patterns and their mapping on the corresponding organisms are shown in Figure 5. Almost all proteins involved in these conserved subnets are well-annotated for *S. cerevisiae*, which facilitates mapping of these annotations to other organisms that share these interaction patterns. The subnet in Figure 5a is a pathway associated with small nuclear ribonucleoprotein complex and is conserved in *D. melanogaster*, *C. elegans*, and *S. cerevisiae*. Proteins Lsm1-7 make up a complex that participates in mRNA degradation and splicing (Bouveret et al., 2000). Proteins Smx3 and Smd2 are sequence homologs of subunits in this complex. The interactions among components of Actin-related protein Arp2/3 complex conserved in *B. taurus*, *H. sapiens*, and *S. cerevisiae*, are shown

TABLE 1. STATISTICS OF MINED PPI NETWORKS AND THE CORRESPONDING ORTHOLOG-CONTRACTED GRAPHS

	PPI network		*Ortholog-contracted graph*		
Organism	*No. of proteins*	*No. of interactions*	*No. of ortholog groups*	*No. of direct interactions*	*No. of indirect interactions*
A. thaliana	288	424	151	133	63
O. sativa	301	340	219	333	217
S. cerevisiae	5157	18,192	1679	5327	43,420
C. elegans	3345	5988	1494	2818	12,968
D. melanogaster	8577	28,829	2849	11,088	65,540
H. sapiens	4541	8577	1940	3868	23,916
B. taurus	195	265	89	126	21
M. musculus	2479	2959	1213	1730	2284
R. norvegicus	696	881	445	714	761

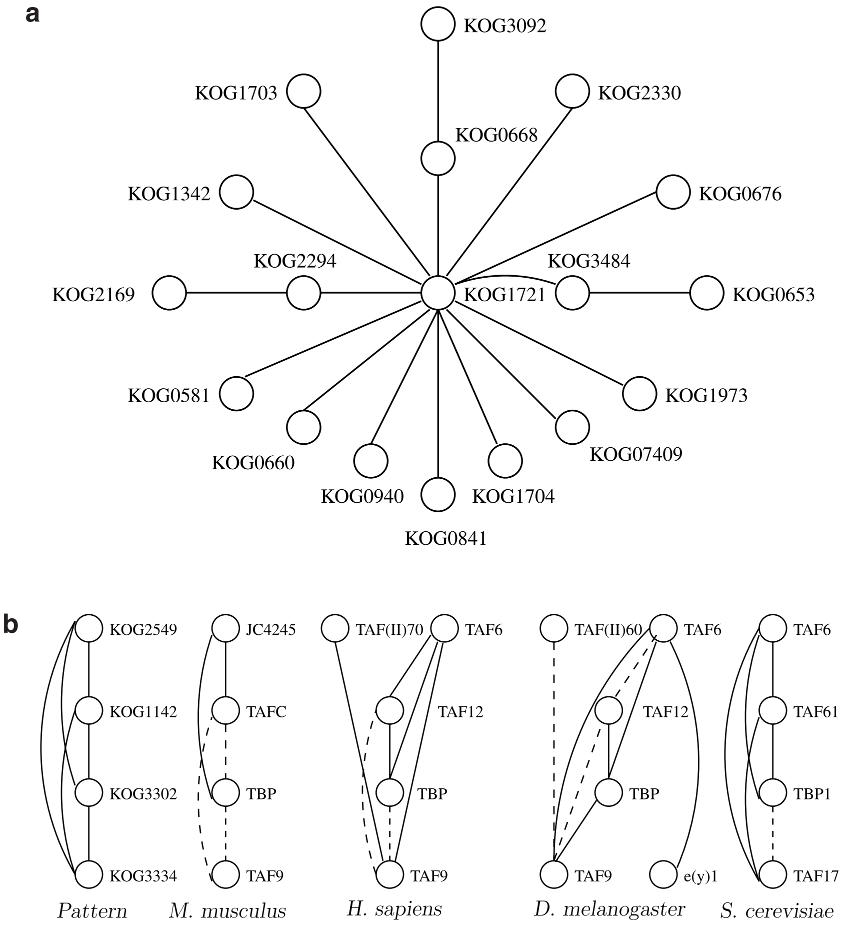

FIG. 4. Frequent interaction patterns that are common to four organisms. (**a**) The frequent interaction pattern that involves interactions of zinc-finger protein, common to *H. sapiens*, *D. melanogaster*, *C. elegans*, and *S. cerevisiae* ($p < 6e - 20$). (**b**) The frequent interaction pattern of TFIID complex and its occurrence in *H. sapiens*, *M. musculus*, *D. melanogaster*, and *S. cerevisiae* ($p < 9e - 51$). Orthologous proteins are horizontally aligned.

in Figure 5b. This complex is involved in actin nucleation. There are seven components known in all for this complex in *S. cerevisiae*, where Arc18 is missing in the conserved subnet (Winter et al., 1999). In the same study, Arc40 is indicated to be essential for viability, which may explain why Arc40 has greater number of interacting partners than the other proteins present in the conserved network. In Figure 5c, two endosomal sorting complexes, ESCRT-II (Vps22, Vsp25, and Vps36) and ESCRT-III (Vps20, Vsp24, and Vps32), are shown to be conserved together in *D. melanogaster*, *S. cerevisiae*, and *H. sapiens*. These two complexes take part in the multivesicular-body pathway and act downstream of another protein complex, ESCRT-I (Hierro et al., 2004). Finally, in Figure 5d, dense interactions between a collection of proteins involved in vesicle transport are detected (Wickner and Haas, 2000). These interactions are conserved in *D. melanogaster*, *S. cerevisiae*, and *R. norvegicus*.

Mining of PPI networks enables not only identification of frequent subgraphs but also phylogenetic analysis of modularity. In Table 2, we list the top eight groups of three organisms based on their shared interactions and subgraphs. While these results may be biased by the varying availability of interaction data for different organisms, they illustrate characteristics of modular phylogeny consistent with sequence-level phylogenetics. For instance, *C. elegans* shares more interactions with *D. melanogaster* and *H. sapiens* than *S. cerevisiae* does, although its available PPI network is less comprehensive. *M. musculus* is always listed with *H. sapiens*, and *R. norvegicus* shares many frequent patterns with *H. sapiens* and *M. musculus* although the PPI data for this organism is very limited. Note that the lack of an interaction pattern in an organism does not necessarily mean that the particular pattern does not exist in that organism, since the

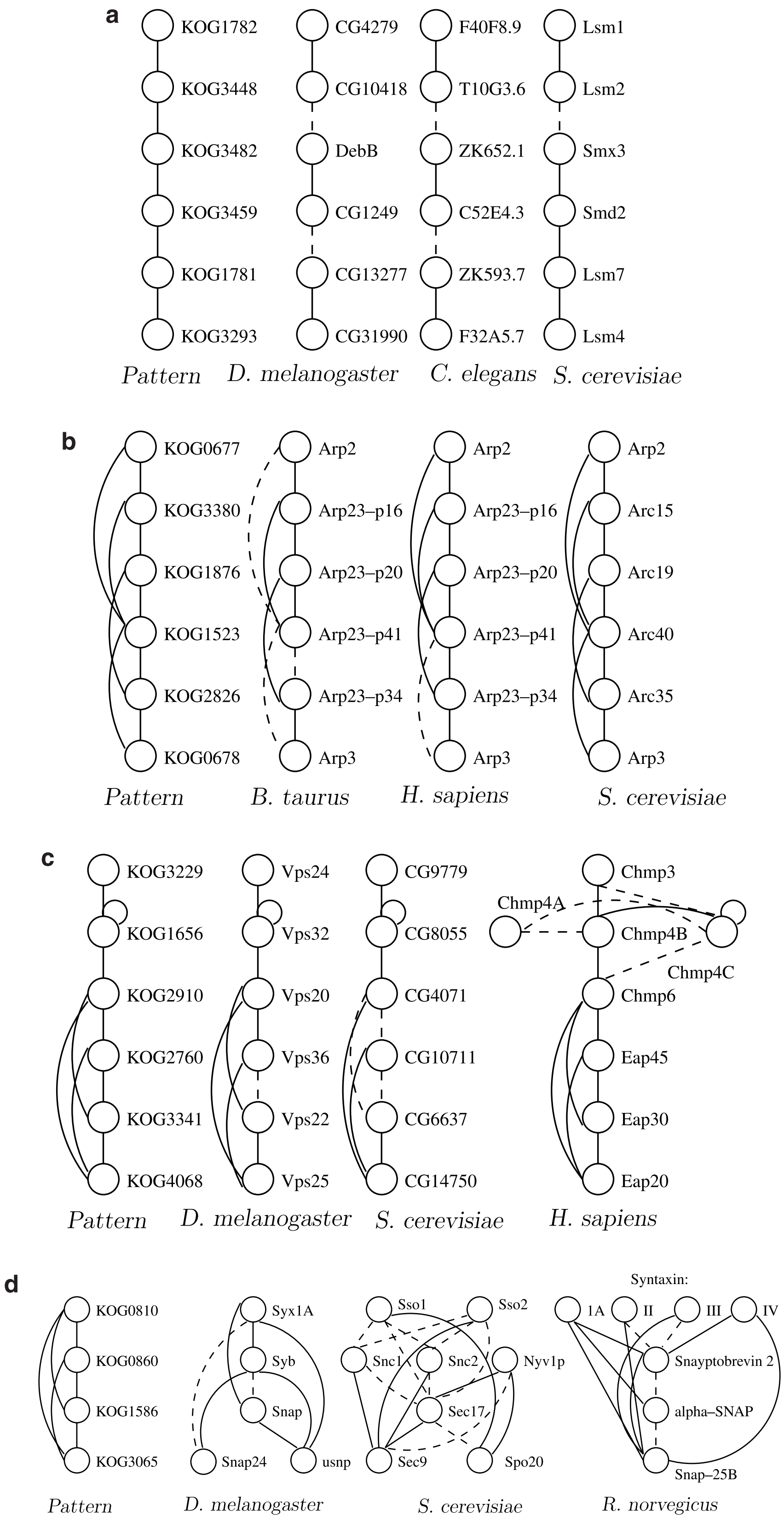

FIG. 5. Sample interaction patterns with frequency three. (**a**) Small nuclear ribonucleoprotein complex ($p < 2e{-}43$). (**b**) Actin-related protein Arp2/3 complex ($p < 9e - 11$). (**c**) Endosomal sorting ($p < 1e - 78$). (**d**) Vesicular transport ($p < 2e - 23$). Orthologous proteins are horizontally aligned.

Table 2. Top Eight Groups of Three Organisms that Contain Most Frequent Connected Subgraphs and Interactions

Organism set	*No. of frequent subgraphs*	*No. of frequent interactions*
C. elegans, D. melanogaster, H. sapiens	8	134
S. cerevisiae, D. melanogaster, H. sapiens	20	126
D. melanogaster, H. sapiens, M. musculus	17	86
S. cerevisiae, C. elegans, D. melanogaster	15	77
S. cerevisiae, C. elegans, H. sapiens	6	50
S. cerevisiae, H. sapiens, M. musculus	10	26
C. elegans, H. sapiens, M. musculus	5	23
H. sapiens, M. musculus, R. norvegicus	10	23

available interaction data is not comprehensive. However, the patterns identified on available data can be used to map known interactions to other species.

4.1.2. Frequent sub-pathways in KEGG metabolic pathways. Using the proposed algorithm, we mine several pathway collections extracted from the KEGG metabolic pathway database. KEGG currently contains a large database of pathway maps for several metabolic processes, including carbohydrate, energy, lipid, nucleotide, and aminoacid metabolism for 157 organisms. We mine several pathways belonging to different metabolisms for different organisms. Sample frequent sub-pathways discovered in pathway collections that belong to glutamate and alanine metabolisms are shown in Figure 6. The nodes of the displayed graphs are labeled by KEGG ID's of enzymes, which can be queried on KEGG web site for detailed information.

We are able to observe fairly large sub-pathways that are frequent. For example, a sub-pathway of glutamate metabolism that contains four nodes and six edges occurs in 45 (29%) of the 155 organisms. This sub-pathway is shown by bold nodes and edges in Figure 6a. It is composed of enzymes glmS (2.6.1.16–glucosamine-fructose-6-phosphate-aminotransferase), guaA (6.3.5.2–GMP synthase), nadE (6.3.5.1–NH(3)-dependent NAD(+) synthetase), and purF (amidophosphoribosyltranferase). In this sub-pathway, all enzymes are related by L-glutamine.

Mining the pathways for different support thresholds allows evaluation of frequent sub-pathways in a multi-level fashion. For instance, when we reduce the required support threshold to 19.3% (30 organisms) for glutamate metabolism, the largest sub-pathway we are able to discover consists of five nodes and 10 edges, which is a supergraph of the previous one. This sub-pathway is shown in the figure by solid nodes and edges. As seen in the figure, this pathway contains enzyme glnA (6.3.1.2–glutamine synthetase), which is also related to the other enzymes by L-glutamine. Further reducing the support threshold to 14.2% (22 organisms), we are able to discover a sub-pathway of six nodes and 13 edges, which is the entire graph shown in the figure. This pathway is also a supergraph of the previous one, with gltX (6.1.1.17–glutamyl-tRNA synthetase) added, which interacts bidirectionally with glnA through L-glutamate. The self-loop for gltX implies that this enzyme takes part in two consecutive reactions, which are part of the observed frequent sub-pathways. The original frequent sub-pathway extracted from this largest frequent ortholog-contracted subgraph is shown in Figure 7a.

In Figure 6b, largest of the frequent sub-pathways that are discovered in alanine-aspartate metabolism for three different levels of support threshold are shown. The bold sub-pathway of five nodes and eight edges occurs in 50 (32.1%) of the 156 organisms, the solid one with five nodes and 11 edges occurs in 30 (19.2%) of the organisms, and the entire graph of six nodes and 16 edges occurs in 18 (11.5%) of the organisms. Note that enzyme purB (4.3.2.2–adenylosuccinate lyase) and its interaction with purA (6.3.4.4–adenylosuccinate synthetase) through adenylosuccinate (N6-(1,2-dicarboxyethyl)-AMP), shown in dotted lines in the figure, is included in the most frequent sub-pathway of alanine-aspartate metabolism but is excluded from the larger sub-pathways of lower frequency, which is interesting to note. The original frequent sub-pathway extracted from the largest frequent ortholog-contracted subgraph is shown in Figure 7b.

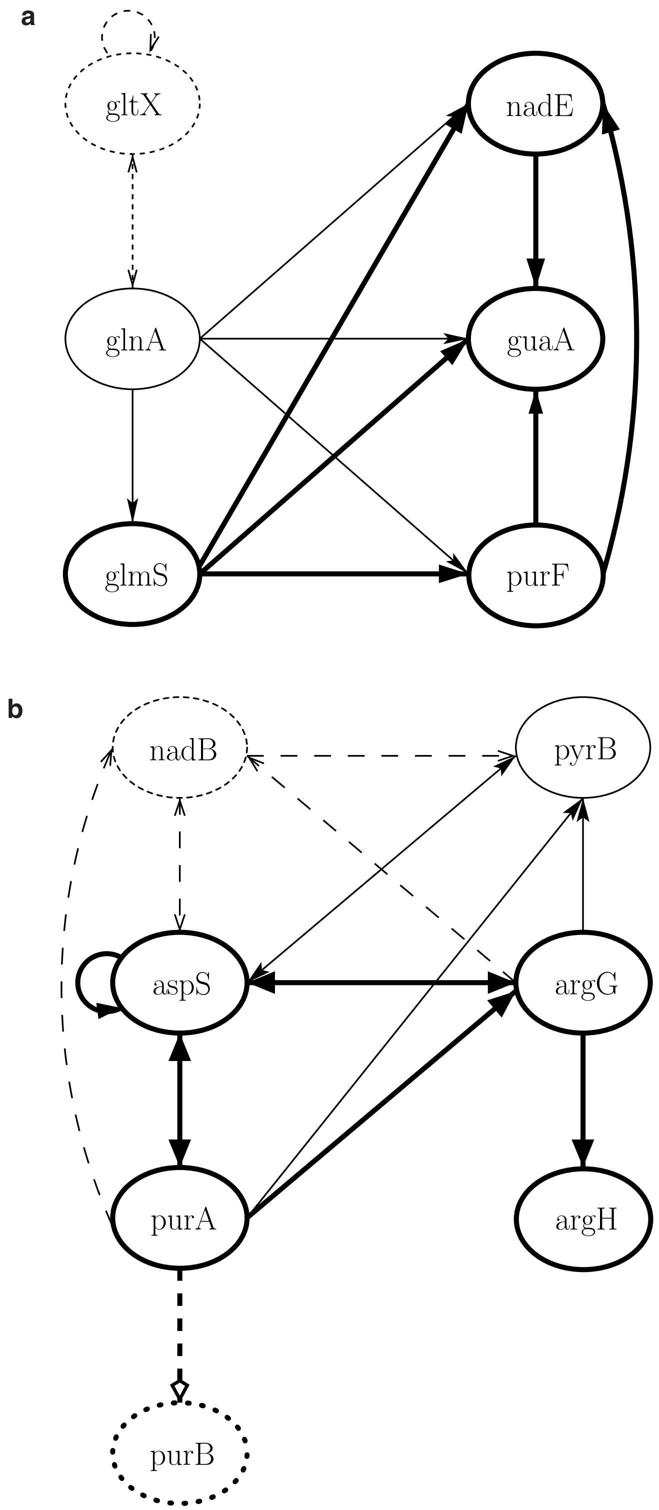

FIG. 6. Frequent edgesets in KEGG metabolic pathways. Frequent subgraphs identified by Mule for different support values on glutamate (**a**) and alanine (**b**) metabolism among 155 and 156 organisms, respectively. Corresponding extracted sub-pathways are shown in Figure 7.

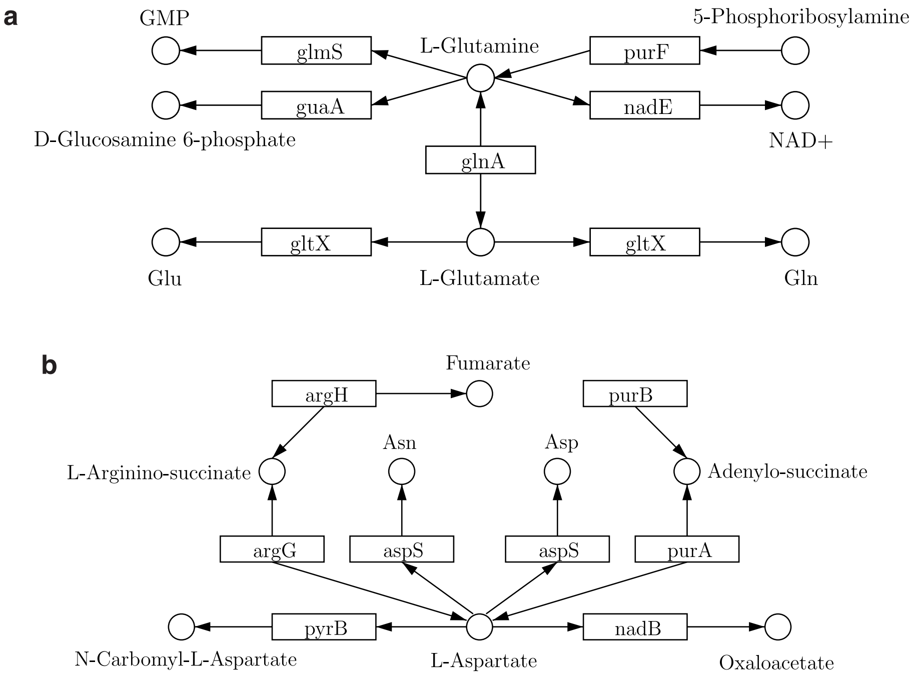

FIG. 7. Sub-pathways extracted from frequent subgraphs discovered by MULE. Frequent sub-pathways extracted from the frequent edgesets shown in Figure 6. (**a**) Glutamate. (**b**) Alanine metabolism.

4.2. Runtime efficiency

In this section, we compare MULE to two existing graph mining algorithms, FSG (Kuramochi and Karypis, 2001) and gSpan (Yan and Han, 2002) to illustrate the effectiveness of node-contraction in terms of runtime performance. All experiments reported in this section are performed on a Pentium-IV 3.0-GHz workstation with 512-MB RAM.

To evaluate runtime efficiency, we rely on metabolic pathways since there is a larger number of available metabolic pathways, making it suitable for illustrating the performance gap between different algorithms. In all of our experiments, we observe that MULE runs much faster than both FSG and gSpan on the graph collections obtained from metabolic pathway datasets. First, we are not able to obtain results from gSpan on the raw directed graphs obtained directly from KEGG metabolic pathways. We suspect that gSpan is not able to respond to these queries because of memory limitations. However, as we illustrate further in this section, gSpan runs very quickly on datasets that are filtered using MULE. The performance comparison of MULE and FSG is shown in Table 3. The runtimes of MULE and FSG along with the number of frequent subgraphs (patterns) and the size of (number of edges in) the largest pattern are shown in the table. As is evident from the figures in the table, MULE runs much faster than FSG by several orders of magnitude. Note that FSG always returns maximal frequent subgraphs. MULE, on the other hand, sometimes returns supersets of frequent subgraphs because of contraction. In our experiments on metabolic pathways, we notice that these supersets are rare and can be easily identified upon examination. Observe that in Table 3, the number of frequent subgraphs discovered by FSG and MULE are the same for all support values in both datasets. This shows that the frequent patterns discovered by the two algorithms correspond to the same set of patterns, while some of these patterns are smaller in MULE, since an edge that actually appears at different locations in the subgraph is contracted into one edge by MULE.

The supersets returned by MULE can be reprocessed through FSG or gSpan and exact frequent subgraphs can be extracted very quickly. This is illustrated in Table 4. In the table, we display the extraction of five largest subgraphs that are discovered by MULE for both datasets. These results show that MULE can be used in a different setup for analysis of biological networks as well. In this setup, a user first mines the graph collection of interest using MULE. Note that, since MULE is fast enough, this can be done repeatedly to tune the minimum support value to obtain the most interesting set of discovered patterns. Upon examination

Table 3. Comparison of Runtime Performances of FSG and Mule on Glutamate and Alanine Metabolic Pathway Datasets

		FSG			*Mule*		
Dataset	*Minimum support (%)*	*Runtime (sec)*	*Largest pattern*	*Number of patterns*	*Runtime (sec)*	*Largest pattern*	*Number of patterns*
Glutamate	20	0.2	9	12	0.01	9	12
	16	0.7	10	14	0.01	10	14
	12	5.1	13	39	0.10	13	39
	10	22.7	16	34	0.29	15	34
	8	138.9	16	56	0.99	15	56
Alanine	24	0.1	8	11	0.01	8	11
	20	1.5	11	15	0.02	11	15
	16	4.0	12	21	0.06	12	21
	12	112.7	17	25	1.06	16	25
	10	215.1	17	34	1.72	16	34

of frequent subgraphs discovered by Mule, the user may choose the patterns of special interest among these. Then, the actual patterns that correspond to these contracted patterns can be extracted by filtering the database and running one of the isomorphism-based graph mining algorithms such as FSG and gSpan. Filtering the graph database reduces the size of the search space substantially in terms of both number and size of graphs to be mined. Indeed, as evident from Table 4, the largest subgraphs that are discovered by Mule are extracted within seconds. In addition, extracting the entire set of frequent subgraphs discovered by Mule takes much less time than mining the original dataset directly, using one of the isomorphism-based algorithms without any preprocessing. As seen in the table, we are able to discover all frequent ($\sigma^* = 8\%$) subgraphs on glutamate pathway collection in 17.8 seconds through preprocessing with Mule followed by isomorphism-based mining with gSpan. Recall that we are not able to mine the original datasets with gSpan alone. Similarly, a combination of Mule and FSG is able to mine this dataset in 101.5 seconds, while FSG alone spends 138.9 seconds to complete the same task. This improvement in runtime (factor of roughly 8) increases rapidly with database size. As databases grow, node contraction is the only known viable approach. In conclusion, while Mule is established as a fast tool for discovering frequent patterns in biological networks in a biologically interpretable fashion, it can also be used to improve other graph mining algorithms. Note also that in the case of protein interaction networks, node contraction is generally necessary for understanding evolutionary relationships.

Table 4. Extraction of Contracted Patterns Discovered by Mule Using FSG and gSpan

Glutamate metabolism, $\sigma^* = 8\%$				*Alanine metabolism, $\sigma^* = 10\%$*			
	Extraction time (sec)				*Extraction time (sec)*		
Size of contracted pattern	*FSG*	*gSpan*	*Size of extracted pattern*	*Size of contracted pattern*	*FSG*	*gSpan*	*Size of extracted pattern*
15	10.8	1.12	16	16	54.1	10.13	17
14	12.8	2.42	16	16	24.1	3.92	16
13	1.7	0.31	13	12	0.9	0.27	12
12	0.9	0.30	12	11	0.4	0.13	11
11	0.5	0.08	11	8	0.1	0.01	8
Total number of patterns: 56				Total number of patterns: 34			
Total runtime of FSG alone: 138.9 sec				Total runtime of FSG alone: 215.1 sec			
Total runtime of Mule+FSG: 0.99+100.5 sec				Total runtime of Mule+FSG: 1.72+160.6 sec			
Total runtime of Mule+gSpan: 0.99+16.8 sec				Total runtime of Mule+gSpan: 1.72+31.0 sec			

4.3. Discussion

MULE is able to detect known functional modules from the interaction networks by exploiting their conservation among different organisms (Figs. 4 and 5). Although our results are limited by the availability of the interaction data, it appears that the conservation of functional modules is a widespread phenomenon observed in numerous cellular activities. Interactions among subunits of protein complexes involved in transcription, mRNA degradation and splicing, actin nucleation, endosomal sorting, and vesicle transport are significantly conserved in yeast and higher eukaryotes, such as humans. This suggests that as more interaction data becomes available, MULE can be used to automatically map functional organization of proteins of a query organism based on the interaction networks of others.

An important problem in large-scale analysis of interaction networks for a growing number of networks arises from the fact that interaction data is noise laden (Titz et al., 2004). Through the ortholog contraction approach in a graph mining based framework, MULE provides robustness and error-correction ability in two ways: (i) interactions that are conserved across a large set of networks are unlikely to arise from noise, (ii) false negatives (existing interactions missing in the data) are likely to be corrected through ortholog contraction, since if two proteins interact, proteins in the same organism that are similar to those in function and sequence are also likely to interact (Wagner, 2001). A limitation of MULE, however, is the modeling of interaction networks using unweighted graphs, whereas weighting interactions provides a more reliable way of accounting for noise (Sharan et al., 2005).

5. CONCLUSION

With the rapidly increasing amount of network and interaction data in molecular biology, the problem of mining patterns, motifs, and modules in biological networks becomes increasingly important. This paper provides a framework for mining biological networks using an innovative graph simplification technique, which leads to efficient graph mining algorithms. The proposed model and algorithm are shown to be well-suited to mining metabolic pathways and protein interaction networks providing interesting results and being able to respond to queries rapidly. It also provides a framework for multi-level analysis of occurrence of interaction patterns in these networks. Our approach can be easily extended to other biological networks as well.

The proposed framework can be further improved by adding flexibility for capturing biologically meaningful information that helps in interpretation of discovered patterns. Finally, the concept of a matching subgraph can be extended to one of an "approximate match." The notions of approximations and distance would need to be formalized before such algorithms can be devised.

ACKNOWLEDGMENTS

This research was supported in part by NIH grant R01 GM068959-01 and NSF grant CCR-0208709.

REFERENCES

Agarwal, R.C., Aggarwal, C.C., and Prasad, V.V.V. 2001. A tree projection algorithm for generation of frequent item sets. *J. Parallel Distrib. Comput.* 61, 350–371.

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules. *Proc. 20th Int. Conf. Very Large Data Bases (VLDB'94)* 487–499.

Altschul, S.F., Madden, T.L., Schffer, A.A., et al. 1997. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.* 25, 3389–3402.

Amy, A.H.Y., Drees, B., Nardelli, G., et al. 2002. A combined experimental and computational strategy to define protein interaction networks for peptide recognition modules. *Science* 295, 321–324.

Auty, R., Steen, H., Myers, L., et al. 2004. Purification of active tfiid from *Saccharomyces cerevisiae*: extensive promoter contacts and co-activator function. *J. Biol. Chem.* 279, 49973–49981.

Bader, G.D., Donalson, I., Wolting, C., et al. 2001. Bind—the biomolecular interaction network database. *Nucleic Acids Res.* 29, 242–245.

Bateman, A., Coin, L., Durbin, R., et al. 2004. The pfam protein families database. *Nucleic Acids Res.* 32, D138–D141.

Berg, J., and Lassig, M. 2004. Local graph alignment and motif search in biological networks. *Proc. Natl. Acad. Sci. USA* 101, 14689–14694.

Bouveret, E., Rigaut, G., Shevchenko, A., et al. 2000. A sm-like protein complex that participates in mRNA degradation. *EMBO J.* 19, 1661–1671.

Burdick, D., Calimlim, M., and Gehrke, J. 2001. MAFIA: a maximal frequent itemset algorithm for transactional databases. *Proc. 17th Int. Conf. Data Eng. (ICDE'01)*.

Chung, F., Lu, L., and Vu, V. 2003. Spectra of random graphs with given expected degrees. *Proc. Natl. Acad. Sci. USA* 100, 6313–6318.

Cook, D.J., and Holder, L.B. 2000. Graph-based data mining. *IEEE Intell. Syst.* 15, 32–41.

Enright, A.J., Dongen, S.V., and Ouzounis, C.A. 2002. An efficient algorithm for large-scale detection of protein families. *Nucleic Acid Res.* 30, 1575–1584.

Gavin, A.C., Bösche, M., Krause, R., et al. 2002. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature* 415, 141–147.

Ghazizadeh, S., and Chawathe, S. 2001. Discovering frequent structures using summaries. Technical report CS-TR-4364. Computer Science Department, University of Maryland, Baltimore.

Goto, S., Bono, H., Ogata, H., et al. 1997. Organizing and computing metabolic pathway data in terms of binary relations. *Pacific Symp. Biocomput.* 175–186.

Gouda, K., and Zaki, M.J. 2001. Efficiently mining maximal frequent itemsets. *IEEE Int. Conf. Data Mining* 163–170.

Han, J.-D.J., Bertin, N., Hao, T., et al. 2004. Evidence for dynamically organized modularity in the yeast protein interaction network. *Nature* 430, 88–93.

Hartwell, L.H., Hopfield, J.J., Leibler, S., et al. 1999. From molecular to modular cell biology. *Nature* 402, C47–C51.

Heger, A., and Holm, L.U. 2003. Exhaustive enumeration of protein domain families. *J. Mol. Biol.* 328, 749–767.

Hierro, A., Sun, J.K.J., Rusnak, A.S., et al. 2004. Structure of the escrt-ii endosomal trafficking complex. *Nature* 431, 221–225.

Hipp, J., Güntzer, U., and Nakhaeizadeh, G. 2000. Algorithms for association rule mining—a general survey and comparison. *ACM SIGKDD Expl. Newslett.* 2, 58–64.

Ho, Y., Gruhler, A., Heilbut, A., et al. 2002. Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature* 415, 180–183.

Hu, H., Yan, X., Huang, Y., et al. 2005. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics* 21, i213–i221.

Huan, J., Wang, W., Bandyopadhyay, D., et al. 2004a. Mining spatial motifs from protein structure graphs. *Proc. 8th Annu. Int. Conf. Res. Comput. Mol. Biol. (RECOMB'04)* 308–315.

Huan, J., Wang, W., Prins, J., et al. 2004b. Spin: Mining maximal frequent subgraphs from graph databases. *Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discov. Data Mining (KDD'04)* 581–586.

Inokuchi, A., Washio, T., and Motoda, H. 2000. An *a priori*-based algorithm for mining frequent substructures from graph data. *Proc. 4th Eur. Conf. Principles Data Mining Knowledge Discov. (PKDD'00)* 13–23.

Ito, T., Chiba, T., Ozawa, R., et al. 2001. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl. Acad. Sci. USA* 98, 4569–4574.

Iuchi, S. 2001. Three classes of c2h2 zinc finger proteins. *Cell. Mol. Life Sci.* 58, 625–635.

Karp, P.D., and Mavrovouniotis, M.L. 1994. Representing, analyzing, and synthesizing biochemical pathways. *IEEE Expert* 11–21.

Kelley, B.P., Sharan, R., Karp, R.M., et al. 2003. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc. Natl. Acad. Sci. USA* 100, 11394–11399.

Koyutürk, M., Grama, A., and Szpankowski, W. 2004. An efficient algorithm for detecting frequent subgraphs in biological networks. *Proc. 12th Int. Conf. Intell. Syst. Mol. Biol. (ISMB'04)* i200–i207.

Koyutürk, M., Grama, A., and Szpankowski, W. 2005. Pairwise local alignment of protein interaction networks guided by models of evolution. *Lect. Notes Bioinform.* 3500, 48–65.

Krishnamurthy, L., Nadeau, J., Özsoyoğlu, G., et al. 2003. Pathways database system: an integrated system for biological pathways. *Bioinformatics* 19, 930–937.

Kuramochi, M., and Karypis, G. 2001. Frequent subgraph discovery. *IEEE Int. Conf. Data Mining* 313–320.

Liébecq, C., ed. 1992. *Biochemical Nomenclature and Related Documents*, 2nd ed. London, Portland Press.

Lotem, E.Y., Sattath, S., Kashtan, N., et al. 2004. Network motifs in integrated cellular networks of transcription-regulation and protein-protein interaction. *Proc. Natl. Acad. Sci. USA* 101, 5934–5939.

Nijssen, S., and Kok, J.N. 2004. A quickstart in frequent structure mining can make a difference. *Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discov. Data Mining (KDD'04)* 647–652.

Olken, F. 2003. Biopathways and protein interaction databases. A Lecture in Bioinformatics Tools for Comparative Genomics: A Short Course.

Oltvai, Z.N., and Barabási, A.L. 2002. Life's complexity pyramid. *Science* 298, 763–764.

Pinter, R.Y., Rokhlenko, O., Yeger-Lotem, E., et al. 2005. Alignment of metabolic pathways. *Bioinformatics* 21, 3401–3408.

Remm, M., Storm, C.E.V., and Sonnhammer, E.L.L. 2001. Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *J. Mol. Biol.* 314, 1041–1052.

Said, M.R., Oppenheim, A.V., and Lauenburger, D.A. 2002. Modeling cellular signal processing using interacting Markov chains. *Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'03)* 6, 41–44.

Sharan, R., Ideker, T., Kelley, B.P., et al. 2004. Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. *8th Int. Conf. Res. Comput. Mol. Biol. (RECOMB'04)* 282–289.

Sharan, R., Suthram, S., Kelley, R.M., et al. 2005. Conserved patterns of protein interaction in multiple species. *Proc. Natl. Acad. Sci. USA* 102, 1974–1979.

Tatusov, R., Fedorova, N., Jackson, J., et al. 2003. The cog database: an updated version includes eukaryotes. *BMC Bioinform.* 4.

Thompson, J.D., Higgins, D.G., and Gibson, T.J. 1994. Clustal-w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* 22, 4673–4680.

Titz, B., Schlesner, M., and Uetz, P. 2004. What do we learn from high-throughput protein interaction data? *Exp. Rev. Prot.* 1, 111–121.

Ullman, J.R. 1976. An algorithm for subgraph isomorphism. *J. ACM* 23, 31–42.

Wagner, A. 2001. The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes. *Mol. Biol. Evol.* 18, 1283–1292.

Washio, T., and Motoda, H. 2003. State of the art of graph-based data mining. *ACM SIGKDD Explor. Newslett.* 5, 59–68.

Wheeler, D.L., Church, D.M., Federhen, S., et al. 2003. Database resources of the national center for biotechnology. *Nucleic Acids Res.* 31, 28–33.

Wickner, W., and Haas, W. 2000. Yeast homotypic vacuole fusion: a window on organelle trafficking mechanisms. *Annu. Rev. Biochem.* 69, 247–275.

Winter, D., Choe, E., and Li, R. 1999. Genetic dissection of the budding yeast arp2/3 complex: a comparison of the *in vivo* and structural roles of individual subunits. *Proc. Natl. Acad. Sci. USA* 96, 7288–7293.

Wuchty, S., Oltvai, Z.N., and Barabási, A.L. 2003. Evolutionary conservation of motif constituents in the yeast protein interaction network. *Nat. Genet.* 35, 176–179.

Xenarios, I., Salwinski, L., Duan, X.J., et al. 2002. Dip: The database of interacting proteins. A research tool for studying cellular networks of protein interactions. *Nucleic Acids Res.* 30, 303–305.

Yan, X., and Han, J. 2002. gSpan: Graph-based substructure pattern mining. *IEEE Int. Conf. Data Mining*, 721–724.

Yan, X., and Han, J. 2003. Closegraph: Mining closed frequent graph patterns. *Proc. 9th ACM SIGKDD Int. Conf. Knowledge Discov. Data Mining (KDD'03)* 286–295.

Yang, J., and Wang, W. 2002. Towards automatic clustering of protein sequences. *Proc. IEEE Comput. Soc. Bioinform. Conf. (CSB'02)* 175–186.

Address correspondence to:
Mehmet Koyutürk
Department of Computer Sciences
Purdue University
West Lafayette, IN 47906

E-mail: koyuturk@cs.purdue.edu