Mining the enriched subgraphs for specific vertices in a biological graph

Mining the Enriched Subgraphs for Specific Vertices in a Biological Graph

Pieter Meysman, Yvan Saeys, Ehsan Sabaghian, Wout Bittremieux, Yves Van de Peer, Bart Goethals, Kris Laukens

Abstract—In this paper, we present a subgroup discovery method to find subgraphs in a graph that are associated with a given set of vertices. The association between a subgraph pattern and a set of vertices is defined by its significant enrichment based on a Bonferroni-corrected hypergeometric probability value. This interestingness measure requires a dedicated pruning procedure to limit the number of subgraph matches that must be calculated. The presented mining algorithm to find associated subgraph patterns in large graphs is therefore designed to efficiently traverse the search space. We demonstrate the operation of this method by applying it on three biological graph data sets and show that we can find associated subgraphs for a biologically relevant set of vertices and that the found subgraphs themselves are biologically interesting.

Index Terms—Subgraph mining, Single graph, Subgroup discovery, Transcriptional Regulatory Network, Protein Graph

1 Introduction

Graph data has become an important resource in various research fields, from social networks in social sciences to transcription regulatory networks in life sciences. There is thus great interest in developing data mining techniques to extract new and useful knowledge from graph data. A common problem is the discovery of frequent subgraphs in a graph data set. This problem can be defined as the identification of those graph patterns that occur more frequently in a given graph data set than a given support threshold. Many solutions exist to address this specific problem, such as GASTON and gSPAN [2], [3]. However often the interesting subgraphs are not simply those that occur most frequently, but those that are most associated with a specific set of vertices. For example, in a biological graph, specific local subgraphs of interest might be those that can be associated with a disease. An example can be found in figure 1. In this example, we can imagine that the circular vertices are regulatory proteins of one type (such as kinases), the square vertices are regulatory proteins of another type (such as transcription factors), and the edges are regulatory interactions from one protein to another. Four of these proteins, namely the blue circular vertices, are known to be involved in disease A, and this disease might impact a specific regulatory activity (represented by the edges). The potential associations of other proteins with this disease are not known, except that most are believed to be irrelevant for disease A. The question is then to find the subgraph motifs that are specific for the disease-associated proteins. In other words, finding those subgraphs that occur more often with the blue vertices than with the white vertices. For this problem, identification of all frequent graphs is unsuitable as there is no guarantee that these are in any way associated with any given set of vertices, i.e. our disease-associated proteins. For example, a regulatory interaction between a disease-associated kinase and a transcription factor is frequent in the graph in figure 1, but the same is true for interactions with kinases irrelevant for disease A. However the specific pattern where the transcription factor self-regulates only occurs with the disease-associated kinases, which is represented by the subgraph at the bottom of figure 1. Simply finding the frequent subgraphs is therefore not enough in this case. Instead, this problem requires a test to check if these subgraphs are associated with a subset of the vertices. In addition, it requires a dedicated algorithm to prune the search space. The identification of associated subgraphs can then be used to uncover insights into the characteristics of these selected vertices, for example the regulatory interactions that these proteins are involved in, and they could then be used as input for a classification approach, for example to identify other disease-associated proteins. Finding the patterns associated with a subset of the database has been termed subgroup discovery. Here we present an approach to uncover the subgraphs significantly associated with a given set of vertices. Furthermore we demonstrate its potential by applying it to three real biological graph data sets. As far as we are aware, this is the first subgroup discovery approach for subgraph mining in the single graph setting.

- *PM, WB, BG and KL are with the Advanced Database Research and Modeling (ADReM) of the University of Antwerp.*
- *PM, WB and KL are Biomedical informatics research center Antwerpen (biomina).*
- *YS is with the VIB Inflammation Research Center and the Department of Respiratory Medicine of the Ghent University.*
- *ES and YVdP are with Department of Plant Biotechnology and Bioinformatics of the Ghent University and Department of Plant Systems Biology of the VIB.*
- *Correspondence E-mail: pieter.meysman@uantwerpen.be*

- *A preliminary version of this article appeared as "Discovery of Significantly Enriched Subgraphs Associated with Selected Vertices in a Single Graph", in the Proceedings of the 15th International Workshop on Data Mining in Bioinformatics (BioKDD'15) [1].*

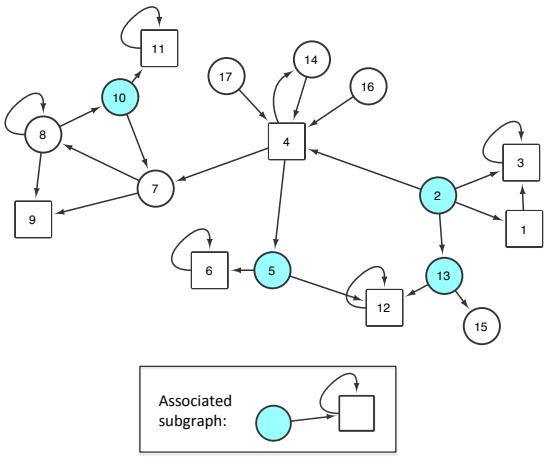

Fig. 1. Example graph data set with the vertex shape denotes its label (square or circle) and the selected vertices are colored in blue. The bottom of the figure includes the detected subgraph associated with the blue vertices.

2 Related work

As mentioned in the introduction, there are several subgraph mining algorithms, tools and methods available for a variety of problems. A subgraph can be considered as a pattern or motif of vertices and edges that occurs frequently in a single or multiple graphs. The subgraphs of interest are then typically those that occur more frequently than a predefined support value. Several subgraph mining algorithms make use of the Apriori principle to prune the search space of frequent subgraphs [4], [5]. Thus, if the frequency of a subgraph is found to be below the support threshold, any other subgraph that contains this subgraph will never be able to pass the support threshold and thus need not be considered. Two types of approaches for traversing the search space can be distinguished here; a breadth-first approach where the frequency of all subgraphs of a given size will be enumerated before extending them, and a depth-first approach where the subgraphs are extended along a branch until they no longer exceed the support threshold. The advantage of breadth-first is that it is more efficient in pruning the candidate tree. However depth-first has a much higher memory efficiency as it only keeps the subgraphs within the same lattice in memory. Further one can make a distinction based on the graph data set structure. Some subgraph miners search frequent subgraphs across different graphs so that the frequency is defined as the number of graphs that contain the subgraph, while subgraph miners that search a single graph will define the frequency as the number of occurrences of the subgraph within that graph [6], [7]. Examples for the single-graph setting as is relevant for this paper, include SEuS, SUBDUE and FSG [8], [9], [10]. In each case, the most intensive step of the algorithm is to check the frequency of a subgraph within the larger graph. Indeed subgraph matching is known to be NP-complete [11]. Therefore the number of subgraphs to be tested must be kept to a minimum to create an efficient algorithm. This includes pruning the search space but also the removal of subgraphs that are identical to each other and thus redundant. A common technique to avoid testing the same subgraph multiple times is to create a canonical representation of each subgraph [2], [10]. Each subgraph is represented by a unique string, vector of edges or an adjacency matrix. The canonical representation is therefore a unique code for this subgraph irrespective of the ordering of the vertices or edges in the graph. Redundant subgraphs can then easily be pruned as they will have the same canonical representation.

The significance of a subgraph in a graph data set can be defined in many different ways. One common definition is that subgraphs are significant if they have a higher frequency than expected from random networks, which have been generated based on a specific network model. Tools to find such subgraphs will often subsample the graphs to speed up the algorithm, and examples include Mfinder, Pajek and FANMOD [12], [13], [14]. While these tools do calculate an enrichment of subgraphs, these are not in any way associated with a specific set of vertices as we propose. There is therefore still no guarantee, that even though these subgraphs are more common in the network than randomly expected, that they have any relation to a set of selected vertices. As far as we are aware, the only subgraph miner for subgroup discovery is iSubgraph [15]. However this algorithm attempts to find subgroups across graphs, not vertices, based on occurring subgraphs. It is therefore only applicable to a setting with multiple graphs and cannot specifically target any relevant selection of vertices.

There are conceptual similarities between the problems that can be addressed with the presented associated subgraph mining approach and pathway or gene ontology enrichment analysis, which is a prevalent procedure within life science research. In the latter case this involves a search for classes or annotations, such as gene ontology terms or pathway assignments, that are significantly associated with a set of biological entities, such as genes or proteins, compared to a given background, typically the entire genome of an organism [16], [17]. As we demonstrate in our case studies, the presented subgraph mining method is able to find the associated subgraphs for a set of biological entities within a biological graph. Thus instead of finding associated annotations, the approach finds associated subgraphs which can include relevant biological annotation labels. The presented method can be seen as an extension of pathway or gene ontology enrichment analysis towards the network context. The underlying statistics used in our algorithm are similar to some of those that have been used within pathway analysis [18]. The presented approach and accompanying implementation has therefore the potential to be a valuable addition to the toolbox for life science research, as network data and network analysis are becoming a common staple for understanding and interpreting biological/biomedical data [19].

3 Definitions and notation

We will start by defining the different terms that will be used in the presented algorithm.

The graph data set subjected to mining is defined as $D = (V, E)$, where V is the set of vertices and $E \subseteq V \times V$ the edges, with a vertex label assignment $\ell \subseteq V \times L$ where

L is the set of all possible labels that can be assigned to vertices. The labeling assignment defines the set of labels $(l_{i1}, ..., l_{in})$ that are assigned to any vertex v_i. For the purposes of this description, we will assume that graph D is directed. Note that the presented algorithm will also work on an undirected graph by assuming that each edge $e_i = (v_i, v_j) = (v_j, v_i)$. Also the graph D need not be fully connected for the operation of the algorithm.

The set of *selected* vertices is defined as $N \subseteq V$ within the graph, which contains the vertices for which we wish to find associated subgraphs.

A subgraph pattern can be defined as $P = (V_P, E_P)$, where V_P is the set of vertices of P and $E_P \subseteq V_P \times V_P$ the edges of P, with a vertex label assignment $\ell_P \subseteq V_P \times L$. While it is theoretically and practically possible to do otherwise, it is often useful to limit each vertex in the subgraph to a single vertex label, as done in the provided implementation and the case studies. This makes sure that no labels are introduced in the subgraph that have no or little matches with the true graph. Critical for the associated subgraph mining search, is the explicit definition of a single vertex $v_{PR} \in V_P$ as the *source vertex* of the subgraph pattern P. This is where our core subgraph matching procedure differs from other approaches. Note that this source vertex is set when a single edge candidate subgraph is generated and it remains the same vertex when extending the graph. There is no explicit constraint on which vertices may be a source vertex. Theoretically for a single subgraph structure there is a candidate subgraph for each vertex as a source vertex, excluding any symmetrical positions. Each of these candidates is thus considered as a separate subgraph that must be tested, even if the subgraph structure is the same, so that the enrichment analysis accounts for the position of the source vertex, which is often biologically relevant.

An instance of a pattern P is defined as the subgraph $G \subseteq D$ if P and G are isomorphic with respect to the edges and vertices labels of each. Thus there exists a mapping function μ which uniquely maps the edges and vertices of G onto P with conservation of the labels, so that $\forall v \in V_P, \ell_P(v) \subset \ell(\mu(v))$ in multiple-label mode or $\forall v \in V_P, \ell_P(v) = \ell(\mu(v))$ in single-label mode. The corresponding vertex $v_{GR} \in V$, so that $\mu(v_{GR}) = v_{PR}$, is defined as the source vertex of the subgraph instance G. The collection of all source vertices in D for the pattern P, i.e. the collection of all v_{GR}, is denoted by R_G.

Each subgraph pattern P and its instances will be described by a canonical string S, so that isomorphic patterns have the same canonical string. The definition of the canonical string encoding can be found in section 4.3.

3.1 Support measure

The total frequency, $Freq_T$, of a given subgraph pattern is defined as equal to $|R_G|$. Thus this corresponds to the total number of vertices, irrespective if they are part of N or not, that are a source vertex for a subgraph instance of pattern P.

The subgroup frequency, $Freq_S$, of a subgraph pattern is defined as equal to $|R_G \cap N|$. Thus, this corresponds to the number of selected vertices in N that are a valid source vertex for the subgraph instance.

Defining the support based on the number of source vertices matching the given subgraph pattern avoids the problem common to single graph subgraph mining where the support does not fit the Apriori criterion due to the occurrence of symmetrical patterns [20]. A vertex can never become a source vertex for a subgraph if it is not a source vertex for any subgraphs that it contains.

3.2 Interestingness measure

If more selected vertices N are source vertices for pattern P than expected when compared to the non-selected vertices $V \setminus N$, we will define the pattern P as associated with, i.e. *enriched* in, the vertices N. This measure must account for the number of selected vertices N and the number of vertices in the graph V that could possibly serve as a source vertex for the subgraph pattern. This is equivalent to the problem in statistics of random sampling without replacement. This concept is best represented by blindly selecting marbles out of a jar that contains a fixed amount of red and blue marbles without putting them back. If we draw a fixed number of marbles at random, we can model the probability distribution of drawn red and blue marbles by the hypergeometric distribution. In this case, the jar with marbles represents all the vertices in the graph and the two colors represent being either a selected vertex or not. The *drawing* of the vertices is then provided by the source vertices of the subgraph R_G. Thus the probability of observing a $Freq_S = X$ is given by:

$$P(Freq_S = X) = \frac{\binom{|N|}{X}\binom{|V|-|N|}{Freq_T - X}}{\binom{|V|}{Freq_T}} \tag{1}$$

The probability of observing a value of $Freq_S$ or higher is then given by the upper cumulative probability:

$$P(Freq_S \geq X) = \sum_{i=X}^{\min(|N|, Freq_T)} P(Freq_S = i) \tag{2}$$

The value $P(Freq_S \geq X)$ thus gives the probability that we find X or more selected vertices as source vertices under the assumption of a uniform distribution of source vertices. This value is commonly known as the *P-value* for the statistical test of enrichment and this is the measure that we will use to define significant subgraphs. The lower the P-value is, the less likely that one would find that number of selected vertices N or more as source vertices R_G if the tested subgraph was independent from these vertices. Therefore if the P-value of a subgraph is sufficiently low, we can claim significant enrichment of the source vertices in the selected vertices and therefore this subgraph pattern can be seen as associated with these selected vertices. The cut-off for the significant P-value is defined as a threshold P_{MAX} so that all subgraph patterns for which $P(Freq_S \geq X) \leq P_{MAX}$ holds true are regarded as interesting and returned. The P-value corresponds to the probability of an associated subgraph being a false positive value and thus can be configured to make the test more stringent. Typical values for the P-value are between 0.1 and 0.01, with respectively a one-in-ten or a one-in-a-hundred chance of a false positive. However as there will be a test performed for each subgraph, we need

to correct the P-value for multiple tests. For this reason, we introduce a Bonferroni correction, a well known multiple testing correction, by dividing P_{MAX} by the number of subgraphs tested prior to checking the significance of patterns [21]. This is the most strict multiple testing correction that can be applied, which will limit the search space and the number of significant patterns to exclude a large fraction of the false positives. Other less conservative multiple corrections can also be used in this context, such as the Benjamini-Hochberg correction [21]. Our choice for the Bonferroni correction is to demonstrate that the algorithm is able to find true positive significantly associated subgraphs across several datasets in the most stringent of circumstances.

4 Significant Subgraph Algorithm

In this section, we introduce an algorithm to find significantly associated subgraphs with a set of vertices within a graph. Three important sections of the algorithm are first described prior to providing an overview of the main algorithm.

4.1 Candidate generation

Candidate significant subgraphs are generated according to a depth-first approach. The initial subgraph consists of a single edge and one or two vertices, each of these edges is assigned a label from L. In the case of a single vertex, the edge goes from the vertex to itself as a so-called 'self-edge' or 'self-loop'. Candidate generation is started twice for the two vertex subgraphs, once assigning the vertices with the incoming edge as the source vertex, and once for the vertex with the outgoing edge. If $Freq_S$ passes the pruning threshold, the subgraph is extended with a single edge. Several possibilities for edge addition are explored. Firstly, an edge can be added between any two vertices already present in the subgraph. Secondly, a single vertex can be added to the subgraph with an outgoing edge to one of the vertices already in the subgraph or an incoming edge from one of the vertices in the subgraph. All possibilities are explored, including different labels for the additional vertex. We will refer to the subgraph pattern that was extended as the 'parent pattern' and the new pattern as the 'child pattern'. As long as the pruning threshold is exceeded, the subgraph pattern is extended until it no longer does so or if it reaches a predefined cutoff on the maximum number of edges, E_{MAX}. After the search space has been entirely exhausted for the initial subgraph, a new combination of vertex and edge labels for a single edge is selected until every possible combination of labels with sufficient frequency has been attempted (see section 4.2).

4.2 Candidate pruning

Candidate significant subgraphs cannot be pruned based on their enrichment P-value as this metric does not satisfy the Apriori condition. Child patterns can easily achieve a lower P-value than their parents. However the P-value for each subgraph is primarily dependent of $Freq_S$ and $Freq_T$, all other variables remain constant for each subgraph. Intuitively it can be realized that a subgraph must exceed a certain value with $Freq_S$ to achieve $P(Freq_S \geq X) \leq P_{MAX}$.

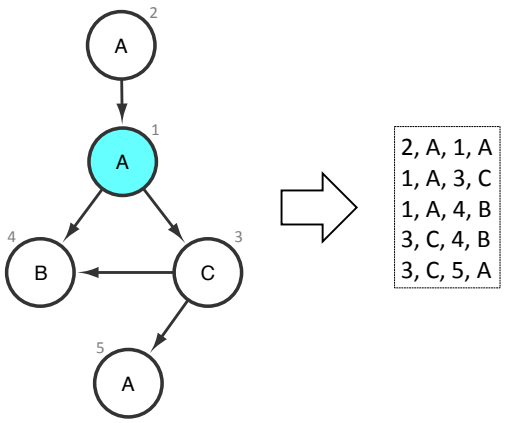

Fig. 2. Example subgraph (left) and its canonical representation (right). First each vertex in the subgraph is assigned an identifier (numbers in grey) based on several criteria, such as distance to the source vertex. The source vertex is always assigned the identifier 1. The edges are then sorted based on the identifiers.

For example, a subgraph where $Freq_S = 0$, i.e. where none of the selected vertices are source vertices, can never be significant. It can be shown that this minimal value of $Freq_S$ always exists and is a value $Freq_{minS} > 0$ so that:

$$P(Freq_{minS}) = \sum_{i=Freq_{minS}}^{|N|} \frac{\binom{|N|}{i}\binom{|V|-|N|}{Freq_{minS}-i}}{\binom{|V|}{Freq_{minS}}} \quad (3)$$

$$P(Freq_{minS}) \leq P_{MAX} \quad (4)$$

$$P(Freq_{minS} - 1) > P_{MAX} \quad (5)$$

Therefore the candidate subgraph tree can be pruned based on $Freq_S > Freq_{minS}$. This is a straightforward frequency count of the number of selected vertices N that are a source vertex of the subgraph pattern P, which satisfies the Apriori condition.

4.3 Subgraph canonical representation

Each subgraph is converted into a canonical representation so that isomorphic subgraphs have the same representation [5]. In this representation, each of the vertices that make up the subgraph is given a unique numeric identifier. Each edge in the representation is then made of two vertex identifiers and two sets of vertex labels if the vertices are labeled. This procedure is based on the canonical representations used by several other graph mining algorithms, which has been shown to avoid redundant subgraph matching [2], [10]. The canonical representation of the subgraph patterns has been modified to account for the presence of a source vertex, which is always explicitly set to an id of 1. Thus for child subgraphs, the source vertex will never change, which is critical for the support measure. Each vertex is then numbered based on a sorting criterion, after which each edge is sorted based on the involved vertex identifiers. Non-source vertices are sorted based on, in the following order, proximity to the source vertex, presence of a self-loop, number of outgoing edges, number of incoming edges and the

vertex label. Edges can then be sorted based on lower vertex identifiers. An example of the canonical representation can be found in figure 2.

4.4 Main algorithm

Algorithm 1 ASSOCIATED_SUBGRAPH_MINER. An algorithm for identifying all subgraph patterns (stored in the set $\mathbb{P}$) significantly associated with the selected vertices N in the graph data set D.

Require: graph D, selected vertices N, all possible labels L, maximum number of edges E_{MAX}, maximum P-value threshold P_{MAX}.

1: $\mathbb{P} = \emptyset, \mathbb{S} = \emptyset, \mathbb{E} = \emptyset$
2: Calculate $Freq_{minS}$ using equation 3
3: **for** each label l_1 of L **do**
4: $\quad \mathbb{E} \leftarrow \mathbb{E} \cup (1, l_1, 1, l_1)$
5: $\quad$ **for** each label l_2 of L **do**
6: $\quad\quad \mathbb{E} \leftarrow \mathbb{E} \cup (1, l_1, 2, l_2)$
7: $\quad\quad \mathbb{E} \leftarrow \mathbb{E} \cup (2, l_1, 1, l_2)$
8: $\quad$ **end for**
9: **end for**
10: **for** each single edge pattern P of $\mathbb{E}$ **do**
11: $\quad$ Subgraph_Search($D, P, N, E_{MAX}, P_{MAX}, Freq_{minS}$)
12: **end for**

In this section we will describe the structure of the main algorithm supporting the subgroup subgraph mining. The initialization procedure is detailed as algorithm 1. In the first step $Freq_{minS}$ is calculated based on the Bonferroni-corrected P_{MAX} to satisfy equation 4 and 5. Each potential initial single edge subgraph is then generated in line 3-9 of algorithm 1 as detailed in section 4.1 and stored in $\mathbb{E}$. Each of these single edge subgraphs is then subjected in turn to algorithm 2. Here line 1-9 evaluates the current pattern and calculates the P-value for significance. In line 11-17, the algorithm will recursively explore each possible extension of the subgraph as long as it exceeds the support threshold and is not larger than the maximum size. Each new subgraph is transformed into a canonical representation using the procedure detailed in section 4.3, to prune redundant subgraphs by comparison to the stored set $\mathbb{S}$. The end result of the entire algorithm will be the set $\mathbb{P}$ containing all significantly associated subgraph patterns for the given selected vertices N.

4.5 Interestingness variants

The interestingness of a subgraph is wholly defined by the enrichment probability as described in the previous section. So any subgraph that occurs more frequently with the selected set of vertices than expected based on the distribution of the subgraph in the entire graph is returned as interesting. However in many real-life applications, this may include several subgraphs that are not relevant to the research question causing unnecessary pattern bloat.

The first variant considers the support of the subgraph within the selected vertices as an equal measure. The enrichment probability only tests significance, however the patterns of interest may be those that also occur at least with a specific frequency. This can be accomplished by manually setting the value for $Freq_{minS}$ instead of using the procedure described in section 4.2. If the manual value is larger than the one that would have been set as the lower bound for enrichment $Freq_{minS}$, the pruning condition still holds. As the orthologous genes case study below demonstrates, this can be as extreme as forcing a match for every selected vertex.

Algorithm 2 SUBGRAPH_SEARCH. A recursive procedure to enumerate the significance of the subgraph P and all its child subgraph patterns. The canonical representation of every tested subgraph is stored in the set $\mathbb{S}$ and every significant subgraph is stored in $\mathbb{P}$.

Require: graph D, subgraph pattern P, selected vertices N, maximum number of edges E_{MAX}, maximum P-value threshold P_{MAX} and the minimum support $Freq_{minS}$ for the selected vertices.

1: Calculate $Freq_S$ for pattern P and vertices N
2: **if** $Freq_S < Freq_{minS}$ **then**
3: $\quad$ **return**
4: **end if**
5: Calculate $Freq_T$ for pattern P in graph D
6: Calculate $P(Freq_S \geq X)$ using equation 2
7: **if** $P(Freq_S \geq X) < P_{MAX}$ **then**
8: $\quad \mathbb{P} \leftarrow \mathbb{P} \cup P$
9: **end if**
10: **if** $size(P) < E_{MAX}$ **then**
11: $\quad$ **for** each child pattern C of P **do**
12: $\quad\quad S \leftarrow$ Canonical(C)
13: $\quad\quad$ **if** $S \notin \mathbb{S}$ **then**
14: $\quad\quad\quad \mathbb{S} \leftarrow \mathbb{S} \cup S$
15: $\quad\quad\quad$ Subgraph_Search($D, S, N, E_{MAX}, P_{MAX}, Freq_{minS}$)
16: $\quad\quad$ **end if**
17: $\quad$ **end for**
18: **end if**

A second variant is the introduction of a background set of vertices, as is common with gene ontology enrichment analysis. In this case, the occurrence of the subgraphs in the selected nodes will not be compared against the occurrence in all nodes of the graph, but a limited subset $B \subset V$. Often this limited subset is a set of vertices with a given label or property that one wishes to compare against. The variable $Freq_T$ is then replaced in the hypergeometric calculation by $Freq_B$ that only counts those subgraph instances with source vertices from the background set, so that:

$$P(Freq_S = X|B) = \frac{\binom{|B|}{X}\binom{|B|-|N|}{Freq_B - X}}{\binom{|B|}{Freq_B}} \quad (6)$$

However for the P-value to remain statistically valid, the background set must include the selected vertices, i.e. $N \subset B$.

A final variant aims at reducing the number of returned subgraph patterns by explicitly considering the child-parent relationship in the enrichment calculation. A common problem with enrichment analysis in a hierarchical setting is that a significantly enriched pattern will cause any derived pattern to also be enriched. For example, if the subgraph

consisting of a single self-edge (1-1) is found to be significantly associated with the selected vertices, all subgraph patterns that contain this edge will also be predisposed to be enriched. This can be seen in the results of many of our use cases, as a single enriched subgraph pattern will explode in a myriad of enriched child patterns that do not have any (or very minimal) difference in $Freq_S$ or $Freq_T$. While these do pass the enrichment cutoff, they are less interesting and confound the interpretation of the results. This can be alleviated by forcing any child subgraph to have a notable change in $Freq_T$ from its parent subgraph without (or less so) a corresponding change in $Freq_S$. Given the design of the algorithm the most elegant way to introduce this change is to integrate it directly into the hypergeometric calculation. Instead of comparing against the entire graph (or selected background), the hypergeometric function can be reconfigured to calculate enrichment based on the matches reported for the parent term ($Freq_{Tp}$,$Freq_{Sp}$):

$$P(Freq_S = X|Freq_{Tp}, Freq_{Sp}) = \frac{\binom{Freq_{Tp}}{X}\binom{Freq_{Tp}-Freq_{Sp}}{Freq_T-X}}{\binom{Freq_{Tp}}{Freq_T}} \quad (7)$$

The calculation remains statistically valid as any match by a child subgraph will always match any parent subgraph, as described in the pruning condition. This equation can be subjected to the same multiple-testing correction and pruning condition as before.

5 Example toy data set

To demonstrate the operation of the method and its use, we applied it to the example data set described in the introduction. This data set consists of a graph with 17 vertices, of which 4 are selected for discovery of associated subgraphs. Each vertex carries one of two possible labels, represented by a circle or a square, as can be seen in figure 1. The structure of the graph was designed so that there is exactly one subgraph with one or two edges significantly associated with the selected vertices. The subgraph mining algorithm was run on this example graph with $P_{MAX} = 0.05$ and $E_{MAX} = 2$. The algorithm correctly identified $Freq_{minS} = 4$, i.e. that all of the selected vertices at minimum have to be source vertices for a subgraph for it to pass P_{MAX} following Bonferroni correction. The only single edge subgraph that passed this criterion is one where a circular vertex has a direct edge to a square vertex: $Freq_S = 4, Freq_T = 9$. This single edge subgraph is still found in 5 other non-selected vertices and scores a P-value of 0.053. It therefore did not pass the significance cutoff P_{MAX}. The extension of this subgraph with a self-edge on the square vertex also passed $Freq_{minS} = 4$. However none of the non-selected vertices were source vertices for this subgraph pattern and therefore $Freq_S = 4, Freq_T = 4$. This subgraph is then found to be significant at a P-value of 4.2×10^{-4}.

6 Experimental applications

We demonstrate the use of this approach on three real biological data sets to find unknown subgraph patterns that are associated with a set of selected vertices. In the first case, we demonstrate the method on a single connected graph with a large fraction of selected vertices (around 1 in 7). In the second and third case, we apply the method to large disconnected graphs with a very small number of selected vertices. The first two cases concern directed graphs involving transcription regulation networks, where each vertex represents a gene and each edge represents a regulatory interaction. Note that these regulatory interactions are always directed and can include self-edges. The last case is a collection of protein graphs, which are undirected and where each edge represents amino acids in close proximity.

6.1 Run times

All experiments were run on a Macbook Pro with 16GB of RAM and a 2.3 GHz processor using the script that is available for download. The run times for each experiment are given in table 1. As with all subgraph miners, the density of the graph is the biggest contributor to the run time. The Yeast graph is smaller but has a much higher average degree than all other tested graphs. Further the Yeast graph has several hubs with hundreds of outgoing edges. Each time the algorithm must compare a subgraph to these hubs, it must iterate over all possible combinations of outgoing edges to check if a match can be found. Interestingness variants as described in 4.5 had very little effect on the run time. The only exception is that if the manually specified minimal support is much higher than that automatically calculated based on the P-value, the search space is drastically reduced as more subgraphs are pruned and the algorithm can traverse all possibilities much quicker.

TABLE 1
Experiment run times

Data set	Vertices	Edges	E_{MAX}	Run time
Yeast	6 402	48 080	3	14h32m24s
Prokaryote	28 609	62 675	5	10m5s
Protein	23 810	60 730	5	1h12m56s

6.2 Duplicated genes in the yeast transcription regulatory network

The first case is a graph representing the transcription regulatory network from baker's yeast as reported by the YEASTRACT database [22]. This graph contains 6 402 genes as vertices, which share 48 080 edges. The selected vertices in this case are genes identified as being retained in duplicate from a recent whole genome duplication [23]. Throughout biological evolution the entire genome (and all of the genes it contains) is on occasion duplicated, so that every gene appears in two copies. Typically the species does not retain both copies but will gradually lose them over the course of many generations. However some genes are known to be retained in duplicate even after several millions of years. There are currently many theories on why such genes are retained, but here we investigate if there are specific subgraph patterns that can be associated with such genes. The most recent whole genome duplication of yeast occurred around 100 million years ago and 900 duplicate genes remain [23]. The selected set of vertices

therefore consists of 900 gene vertices and E_{MAX} was set to 3, which resulted in $Freq_{minS} = 3$. The labels of the graph are derived from the gene ontology assignments available in Uniprot-GO for yeast [24]. To allow for meaningful patterns, we reduced the total gene ontology annotation to 15 terms by traversing the hierarchical tree of these terms [25]. A traditional gene ontology enrichment analysis limited to these 15 terms, found that the duplicated genes were enriched in stress response (P-value = 2.3×10^{-4}), sporulation (3.0×10^{-3}), signal transduction (6.0×10^{-4}) and carbon metabolism (2.4×10^{-5}). These enrichment results match current knowledge on these duplicated genes but they reveal nothing about the positioning of these genes in the regulatory network or the associated network motifs. As some vertices have multiple gene ontology annotations, the subgraph miner was run in the multiple-label mode. This implicitly allows inclusion of vertices without annotation. In a brute force approach without any pruning, we would need to evaluate the enrichment of more than 50 million different subgraphs (the number of possible edge configurations times the number of possible label assignments). However pruning for subgraphs that had at least three selected vertices (duplicated genes) as source vertices reduced the search space to 136 327 subgraphs, of which 24 766 were found to be significantly enriched. However many of the found subgraph patterns showed clear parent-child relationships; i.e. if a parent subgraph was enriched so too were many of its child patterns. To focus on the causal subgraph patterns for enrichment, the algorithm was run in the variant mode that requires enrichment of child pattern matches with regards to the parent matches. This reduced the number of enriched subgraphs to 427, a much more manageable number to survey for biological relevance.

Several examples of enriched subgraphs that remained can be found in figure 3. One-edge patterns made up 133 of these subgraphs. This includes the pattern where an unlabeled source vertex has one outgoing edge, the patterns with an incoming edge and the pattern with a self-edge. In the multiple-label mode of the algorithm, an unlabeled vertex in the subgraph can match any vertex in the graph, as described previously. Thus the duplicated genes are enriched for those that regulate other genes, are regulated by other transcription factors and have self-regulation. The fact that all three possible single edge motifs are enriched, signifies that the duplicated genes occur more often at the center of the regulatory network than on the outside. This implies that most of these will likely have an important function in the regulatory pathways that manage the yeast gene expression, which is in line with current hypotheses. Further the enrichment for self-edges is interesting by itself and has not been noted before as far as we are aware. Self-regulation may be an underlying reason to explain why these genes have been retained in duplicate. As multiple copies of the genes may exist within the genome at the same time, strict self-regulation could play a big role in managing the overall expression levels. The other single-edge patterns mostly concerned various combinations of gene ontology labels.

There were 33 two-edge and 262 three-edge patterns found to be enriched with the parent-child enrichment variant. While the normal version of the subgraph miner found several enriched subgraph patterns of complex regulatory motifs featuring the source vertex as a major regulator (in the set of 24 766 patterns), the variant finds almost none. From the algorithmic perspective, this is because these complex patterns are all derived from the single-edge pattern where the source vertex has an outgoing edge. In the normal operation enrichment of the subgraph matches for the regulatory motifs would be done against the entire graph background, but in the parent-child variant this is checked against the parent matches, i.e. those vertices that have an outgoing edge in this case. This indicates that while the duplicates genes do contribute to complex regulatory systems as regulators, they are not more likely to do so than other regulators. The two-edge and three-edge enriched subgraphs all deal with patterns where the source vertex is regulated. For example, the subgraph with three incoming edges to the source vertex was found to be enriched for the duplicated genes. This indicates that these genes are under tight regulation, even more so than can be expected for all genes that have at least one or two incoming edges (which would have matched the parent subgraphs in this case). Further, as can be seen in figure 3, several subgraphs were enriched that represent a regulatory loop structure, where the regulator and the duplicated gene are both regulated by a third regulator. In addition, the duplicated genes are often downstream from regulators involved in stress response, sporulation, nitrogen- and carbon metabolism. The found motifs match current biological hypotheses that these duplicated genes often have an important regulatory role in the transcription networks, in particular with respect to general metabolic processes and stress responses [23], [26]. However the complex regulatory interactions that regulate the expression of these genes (self-edges, loops, etc.) are unique to this study as no other algorithm exists to investigate this type of pattern.

6.3 Orthologous genes in prokaryotic transcription regulation networks

The second case is a data set featuring the combined transcription regulatory network from seven bacterial organisms with 28 609 vertices and 62 675 edges. While this is a single graph data set, it contains seven independently connected graphs. These graphs were obtained by applying the GENIE3 prediction algorithm [27] on the Colombos expression database [28]. A common life science research question is to investigate the difference and similarities between the regulatory networks of different species, as this can provide new insight into how each has adapted to its environment [29]. The selected vertices were defined as the 10 homologs of the PhoR transcription factor, i.e. the 10 copies of this gene that exist in these seven networks. This transcription factor is present in all seven species and is involved in the regulation of the phosphate homeostasis, which is of critical importance for both the energy levels of the organism and the creation of new molecular compounds, such as DNA and RNA, in living cells. Application of the proposed subgraph mining algorithm in this manner allows us to track the PhoR vertex across different graphs, representing different species, and characterize the subgraphs that occur consistently throughout. However,

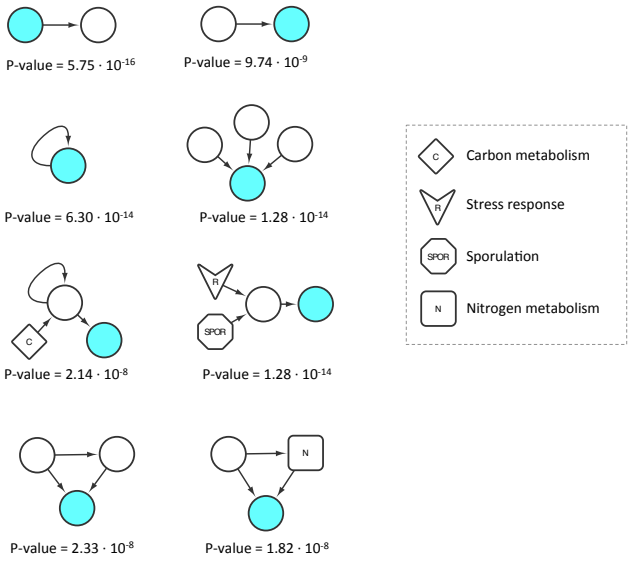

Fig. 3. Selected subgraphs with duplicated genes in the yeast transcription regulatory network. Duplicated genes were set as the selected vertices. Labels are provided by the vertex shape (unlabeled vertices are represented by a circle) and the subgraph source vertices are marked in blue. Reported P-values are the likelihood of depletion based on the frequencies observed for the parent subgraph of the pattern.

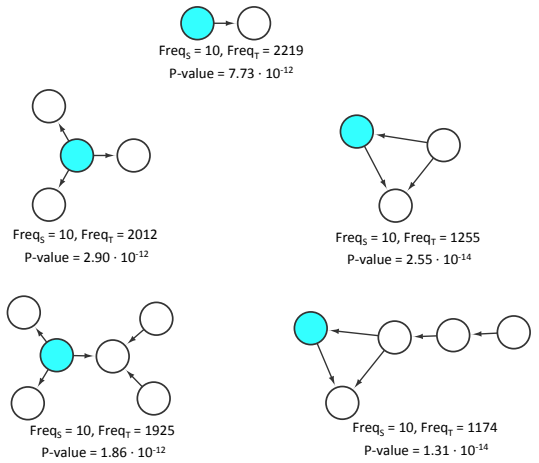

Fig. 4. Selected subgraphs with the P-value for enrichment with the 10 PhoR homologs in the prokaryotic transcription regulation networks as compared to the entire graph. PhoR homologs were set as the selected vertices. The subgraph source vertices are marked in blue.

this problem cannot be addressed with a multiple graph algorithm, despite spanning several graph datasets. This is because the interestingness, and in turn the support, of the subgraph pattern needs to be defined for the selected vertices and not on the graph level, as some graph data sets have more than one PhoR vertex. As we wish to find the subgraphs that can be consistently found with PhoR, we explicitly set $Freq_{minS} = 10$ so that all PhoR copies must be source vertices of the subgraph to be considered. This value is higher than the one generated through the procedure described in section 4.2, but has no further impact on the operation of the algorithm. The significance measure, namely the hypergeometric P-value, does not change as even with $Freq_S = 10$ it is possible to find subgraphs which are not enriched. No labels were included in this graph data set so that the subgraphs are focused on the graph structure.

At $E_{MAX} = 5$, a total of 257 subgraph configurations were checked for enrichment with the PhoR homologs, of which 169 passed the Bonferroni adjusted P-value cut-off. As can be seen in figure 4, only one single edge subgraph was found to be significant, namely the subgraph with an outgoing edge from the source vertex. However larger significant subgraphs do include incoming edges into the source vertex, again demonstrating the notion that a child subgraph can be significantly associated even if the parent subgraph is not. Further as $Freq_S$ is fixed to 10, the P-value is now entirely dependent on $Freq_T$, so that lower values will always correspond to a lower P-value. Other example associated subgraphs include the commonly found feed-forward/back loop in regulatory networks, and subgraphs that include several outgoing edges from the source vertex. These results are in line with the current knowledge on PhoR, namely that it is a transcription factor with many outgoing edges and that it and its targets are frequently under the control of the same global regulators. However the results are different if we start with the knowledge that all PhoR homologs are transcription factors (with at least one outgoing edge). We reran the dataset using the list of known transcription factors in these species as the background set. This means that for the enrichment we no longer compare against the entire graph but only against this set of background vertices. As expected, we no longer find a significant enrichment for the pattern where PhoR has a single outgoing edge. This is because now all background vertices also match this same pattern and it cannot be significantly associated with the PhoR homologs anymore. In addition, all other patterns described before no longer pass the significance threshold with the background set. This indicates that the patterns found to be associated with PhoR in our first analysis are equally present in all transcription factors, or at least in an amount that is not statistically significant less with our strict Bonferroni cut-off. An important criterium for any miner based in statistics is that it should not return anything when there is nothing. Thus in future applications (and the other cases described here), these results are a strong indication that any returned patterns are indeed associated with the selected vertices as long as one considers the correct background and context.

6.4 Manganese binding motifs in peptidase protein structures

The final case concerns a dataset of 72 peptidase protein graphs. These protein graphs are derived from reported molecular structures of peptidase proteins that bind a manganese (Mn) ion from the RCSB Protein DataBase (PDB) [30]. The graph represents the protein molecular structure where the vertices are the amino acid residues that make up the protein and the edges denote the residues whose Cα are within 4 angstrom. The selected vertices in this case are the residues that are reported to bind the Mn-ion. Note that the Mn-ion itself is not part of the graph as we wish to

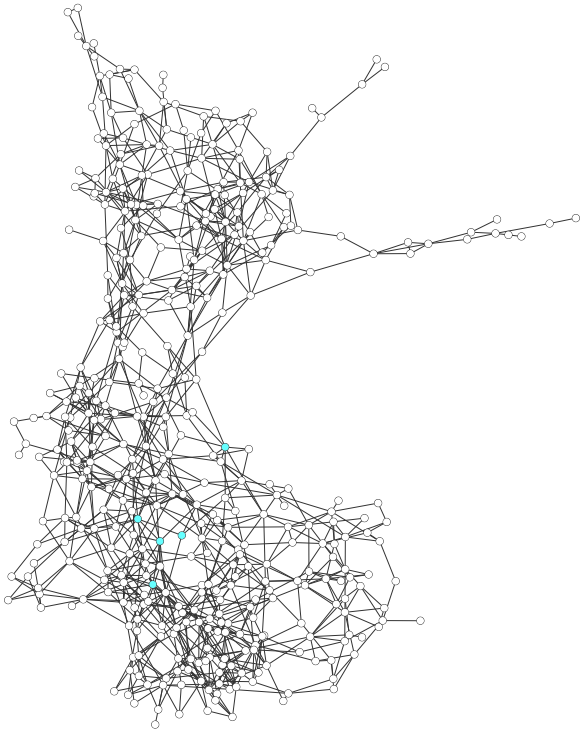

Fig. 5. Protein graph of the molecular structure of a single peptidase (PDB: 1A16). Mn-interacting residues vertices are marked in blue.

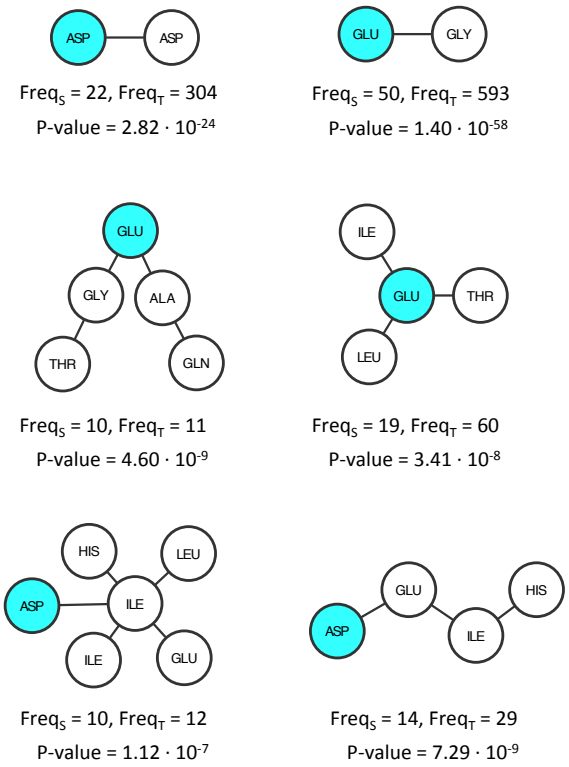

Fig. 6. Significantly enriched subgraphs for Mn-interacting residues. Labels correspond to amino acids (given in the three-letter code). Source residue vertices are marked in blue.

find the patterns in the protein structure itself that allow it to bind the Mn-ion. Again this problem cannot be solved using multigraph approaches as each protein can have multiple bound Mn-ions and each Mn-ion can be bound by multiple residues. The graph includes 60 730 edges and 23 810 vertices, of which 208 bind a Mn-ion and are passed along as the selected vertices.

The subgraph miner was run in single-label mode as each vertex has a single label, namely the corresponding amino acid. This resulted in 2118 different significantly enriched subgraphs (of the 161 388 subgraph patterns that were tested). To reduce this number, we ran the algorithm in the child-parent variant and ended up with 99 subgraphs, which is a much more manageable number for functional analysis. As shown in figure 6, only subgraphs where the source vertex is an aspartate, glutamate or histidine were found to be enriched. This corresponds to previous analyses, which have shown that these three amino acids are solely responsible for the interaction with Mn-ions [31], [32]. While this would normally bias any resulting child subgraphs to also be enriched as long as they have an aspartate, glutamate or histidine as a source vertex, this is not a problem in the child-parent variant. Each subgraph enrichment is calculated based on the frequencies of the parent subgraph. Thus the reported enrichment P-values in figure 6 for two-edge or larger subgraphs are already compared to other subgraph matches with this residue bias. Without any input parameter tuning, the parent-child variant algorithm will automatically return enriched subgraphs accounting without bias for the previously found enrichment. Many of the larger enriched subgraphs feature other instances of aspartate, glutamate or histidine. This makes sense as all these residue will be in close proximity when they all bind the same Mn-ion. Other than these three amino acids, many patterns also feature the small hydrophobic amino acids, leucine, isoleucine and alanine. This might indicate that these Mn-binding sites are often near the hydrophobic core of the protein, or are surrounded by α-helices or β-sheets. As far as we are aware, this is the first attempt at data mining in the three-dimensional structure beyond the interacting amino acids themselves for Mn-binding patterns (represented here in a graph structure). The found patterns could be useful as input to train a classifier to detect other Mn-binding regions.

7 Conclusion

In this paper, we have presented a novel type of subgraph mining approach for subgroup discovery that can be used to extract specific subgraph patterns related to a set of selected vertices in a single graph. This approach has been shown to be useful on three biological data sets for uncovering novel patterns that are biologically relevant. In addition, other common life science problems can also be tackled with this method. A common use case for enrichment analysis is where a biological experiment results in a set of differentially expressed genes, proteins or metabolites, which are then compared to a background set to search for common functions, properties or regulatory modules that are significantly associated with this set. This method is a nice complement to the existing enrichment-analysis

methodologies, especially for those who want to integrate a network-component, such as regulatory networks, protein-protein interaction networks or metabolic networks. However the presented approach is generic and not limited to a biological setting. It therefore hold great promise to discover patterns associated with fraudulent behavior in financial graphs, or patterns associated with deviant behavior in social networks. In summary, any instance where the subgraphs of interest are those that should be associated with a specific set of given vertices.

An implementation of the significant subgraph miner for both directed and undirected labeled graphs can be found on the ADReM website (*http://adrem.ua.ac.be/sigsubgraph*) and tracked in a code repository (*https://bitbucket.org/pmeysman/sigsubgraphminer*). This implementation allows the user to set any of the introduced interestingness variants (default, at a fixed support, with a background set and parent-child enrichment) and includes the featured toy example data set.

8 Future work

Several refinements and extensions of the presented approach are still possible.

Firstly, the underlying algorithm that is used in this study can still be optimized. Especially as much of the computation time is spent enumerating subgraph matches in the complete graph, which is a similar task addressed by other subgraph mining approaches and algorithms. The presented algorithm, and accompanying implementation, does already use several common subgraph mining principles to prune the candidate tree and speed up subgraph matching problems, such as support pruning and canonical labeling. However other optimizations and approaches have been developed for subgraph mining, which can likely be applied in this setting, such as tree-based data structures to store and prune candidates. The reason that these approaches could not be immediately applied here is because the candidate generation problem differs from common approaches as the subgraphs are defined with an explicit source vertex. This means for example that what is normally considered as a single unique subgraph may be two or more unique subgraphs in this setting. Moreover the enumeration strategy is also impacted by the definition of a source vertex; what might be considered as a hit from normal approaches, may not be appropriate for this setting and vice versa. Research is therefore still needed to adapt these optimization strategies to this approach.

Secondly, dedicated variants on the approach can be designed to solve specific problems. For example, the subgraph search space could be limited to subgraphs of a given type, such as depleted instead of enriched subgraphs, and this in turn could lead to different and more efficient pruning conditions.

Acknowledgments

The authors wish to thank Stefan Naulaerts, Cheng Zhou, Tayena Hendrickx and Aida Mrzic for the helpful discussions and insights. This work was supported by the Fund for Scientific Research - Flanders (FWO-Vlaanderen) project "Evolving graph patterns". YVdP acknowledges support from Ghent University [Multidisciplinary Research Partnership "Bioinformatics: from nucleotides to network"] and from the European Union Seventh Framework Programme [FP7/2007-2013] under ERC Advanced Grant Agreement no. 322739 - DOUBLE-UP.

References

[1] P. Meysman, Y. Saeys, E. Sabaghian, W. Bittremieux, Y. V. D. Peer, B. Goethals, and K. Laukens, "Discovery of Significantly Enriched Subgraphs Associated with Selected Vertices in a Single Graph," in *Proceedings of the 14th International Workshop on Data Mining in Bioinformatics (BioKDD'15)*, Sydney, Australia, 2015.

[2] X. Yan and J. Han, "gSpan: graph-based substructure pattern mining," in *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* IEEE Comput. Soc, 2002, pp. 721–724. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=1184038

[3] S. Nijssen and J. N. Kok, "The Gaston Tool for Frequent Subgraph Mining," *Electronic Notes in Theoretical Computer Science*, vol. 127, no. 1, pp. 77–87, Mar. 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1571066105001064

[4] A. Inokuchi, T. Washio, and H. Motoda, "An Apriori-based Algorithm for Mining Frequent Substructures from Graph Data," *Princ Data Min Knowl Discov*, vol. 1910, pp. 13–23, 2000. [Online]. Available: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.36.2485

[5] M. Kuramochi and G. Karypis, "Frequent subgraph discovery," in *Proceedings 2001 IEEE International Conference on Data Mining*. IEEE Comput. Soc, 2001, pp. 313–320. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=989534

[6] C. Jiang, F. Coenen, and M. Zito, "A survey of frequent subgraph mining algorithms," *The Knowledge Engineering Review*, vol. 28, no. 01, pp. 75–105, Nov. 2012. [Online]. Available: http://journals.cambridge.org/abstract_S0269888912000331

[7] S. Naulaerts, P. Meysman, W. Bittremieux, T. N. Vu, W. Vanden Berghe, B. Goethals, and K. Laukens, "A primer to frequent itemset mining for bioinformatics." *Briefings in bioinformatics*, vol. 16, no. 2, pp. 216–31, Mar. 2015. [Online]. Available: http://bib.oxfordjournals.org/content/16/2/216

[8] D. J. Cook and L. B. Holder, "Substructure Discovery Using Minimum Description Length and Background Knowledge," *Journal of Artificial Intelligence Research*, pp. 231–255, 1994. [Online]. Available: http://www.jair.org/papers/paper43.html

[9] S. Ghazizadeh and S. S. Chawathe, "SEuS: Structure Extraction Using Summaries," in *Proceedings of the 5th International Conference on Discovery Science*. Springer-Verlag, Nov. 2002, pp. 71–85. [Online]. Available: http://dl.acm.org/citation.cfm?id=647859.736135

[10] M. Kuramochi and G. Karypis, "An efficient algorithm for discovering frequent subgraphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1038–1051, Sep. 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1316833

[11] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the third annual ACM symposium on Theory of computing - STOC '71*. New York, New York, USA: ACM Press, May 1971, pp. 151–158. [Online]. Available: http://dl.acm.org/citation.cfm?id=800157.805047

[12] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon, "Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs." *Bioinformatics (Oxford, England)*, vol. 20, no. 11, pp. 1746–58, Jul. 2004. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/20/11/1746.short

[13] V. Batagelj and A. Mrvar, *Pajek: analysis and visualization of large networks*, 2004. [Online]. Available: http://link.springer.com/content/pdf/10.1007/978-3-642-18638-7_4.pdf

[14] S. Wernicke and F. Rasche, "FANMOD: a tool for fast network motif detection." *Bioinformatics (Oxford, England)*, vol. 22, no. 9, pp. 1152–3, May 2006. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/22/9/1152.short

[15] B. Ozdemir, W. Abd-Almageed, S. Roessler, and X. W. Wang, "iSubgraph: integrative genomics for subgroup discovery in hepatocellular carcinoma using graph mining and mixture models." *PloS one*, vol. 8, no. 11, p. e78624, Jan. 2013. [Online]. Available: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0078624

[16] P. Khatri, M. Sirota, and A. J. Butte, "Ten years of pathway analysis: current approaches and outstanding challenges." *PLoS computational biology*, vol. 8, no. 2, p. e1002375, Jan. 2012. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3285573

[17] K. Laukens, S. Naulaerts, and W. Vanden Berghe, "Bioinformatics approaches for the functional interpretation of protein lists: from ontology term enrichment to network analysis." *Proteomics*, vol. 15, no. 5-6, pp. 981–96, Nov. 2014. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/25430566

[18] I. Rivals, L. Personnaz, L. Taing, and M.-C. Potier, "Enrichment or depletion of a GO category within a class of genes: which test?" *Bioinformatics (Oxford, England)*, vol. 23, no. 4, pp. 401–7, Feb. 2007. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/17182697

[19] P. Meysman, K. Titeca, S. Eyckerman, J. Tavernier, B. Goethals, L. Martens, D. Valkenborg, and K. Laukens, "Protein complex analysis: From raw protein lists to protein interaction networks." *Mass spectrometry reviews*, dec 2015. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/26709718

[20] N. Vanetik, S. E. Shimony, and E. Gudes, "Support measures for graph data," *Data Mining and Knowledge Discovery*, vol. 13, no. 2, pp. 243–260, May 2006. [Online]. Available: http://link.springer.com/10.1007/s10618-006-0044-8

[21] J. H. McDonald, *Handbook of biological statistics.*, vol. 2 ed. Baltimore: Sparky House Publishing, 2009.

[22] M. C. Teixeira, P. T. Monteiro, J. F. Guerreiro, J. P. Gonçalves, N. P. Mira, S. C. Dos Santos, T. R. Cabrito, M. Palma, C. Costa, A. P. Francisco, S. C. Madeira, A. L. Oliveira, A. T. Freitas, and I. Sá-Correia, "The YEASTRACT database: an upgraded information system for the analysis of gene and genomic transcription regulation in Saccharomyces cerevisiae." *Nucleic acids research*, vol. 42, no. 1, pp. D161–6, Jan. 2014. [Online]. Available: http://nar.oxfordjournals.org/cgi/content/long/42/D1/D161

[23] M. Kellis, B. W. Birren, and E. S. Lander, "Proof and evolutionary analysis of ancient genome duplication in the yeast Saccharomyces cerevisiae." *Nature*, vol. 428, no. 6983, pp. 617–24, Apr. 2004. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/15004568

[24] E. C. Dimmer, R. P. Huntley, Y. Alam-Faruque, T. Sawford, C. O'Donovan, M. J. Martin, B. Bely, P. Browne, W. Mun Chan, R. Eberhardt, M. Gardner, K. Laiho, D. Legge, M. Magrane, K. Pichler, D. Poggioli, H. Sehra, A. Auchincloss, K. Axelsen, M.-C. Blatter, E. Boutet, S. Braconi-Quintaje, L. Breuza, A. Bridge, E. Coudert, A. Estreicher, L. Famiglietti, S. Ferro-Rojas, M. Feuermann, A. Gos, N. Gruaz-Gumowski, U. Hinz, C. Hulo, J. James, S. Jimenez, F. Jungo, G. Keller, P. Lemercier, D. Lieberherr, P. Masson, M. Moinat, I. Pedruzzi, S. Poux, C. Rivoire, B. Roechert, M. Schneider, A. Stutz, S. Sundaram, M. Tognolli, L. Bougueleret, G. Argoud-Puy, I. Cusin, P. Duek-Roggli, I. Xenarios, and R. Apweiler, "The UniProt-GO Annotation database in 2011." *Nucleic acids research*, vol. 40, no. D1, pp. D565–570, Nov. 2011. [Online]. Available: http://nar.oxfordjournals.org/cgi/content/abstract/40/D1/D565

[25] S. Raychaudhuri, J. T. Chang, P. D. Sutphin, and R. B. Altman, "Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature." *Genome research*, vol. 12, no. 1, pp. 203–14, Jan. 2002. [Online]. Available: http://genome.cshlp.org/content/12/1/203.short

[26] G. Musso, M. Costanzo, M. Huangfu, A. M. Smith, J. Paw, B.-J. San Luis, C. Boone, G. Giaever, C. Nislow, A. Emili, and Z. Zhang, "The extensive and condition-dependent nature of epistasis among whole-genome duplicates in yeast." *Genome research*, vol. 18, no. 7, pp. 1092–9, Jul. 2008. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2493398\&tool=pmcentrez\&rendertype=abstract

[27] V. A. Huynh-Thu, A. Irrthum, L. Wehenkel, and P. Geurts, "Inferring regulatory networks from expression data using tree-based methods." *PloS one*, vol. 5, no. 9, p. e12776, Jan. 2010. [Online]. Available: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0012776

[28] P. Meysman, P. Sonego, L. Bianco, Q. Fu, D. Ledezma-Tejeida, S. Gama-Castro, V. Liebens, J. Michiels, K. Laukens, K. Marchal, J. Collado-Vides, K. Engelen, and F. Qiang, "COLOMBOS v2.0: An ever expanding collection of bacterial expression compendia," *Nucleic Acids Res.*, vol. 42, no. 1, pp. D649–53, Jan. 2014. [Online]. Available: http://nar.oxfordjournals.org/content/42/D1/D649.short

[29] P. Meysman, A. Sanchez-Rodríguez, Q. Fu, K. Marchal, and K. Engelen, "Expression divergence between Escherichia coli and Salmonella enterica serovar Typhimurium reflects their lifestyles." *Molecular biology and evolution*, vol. 30, no. 6, pp. 1302–1314, feb 2013. [Online]. Available: http://dx.doi.org/10.1093/molbev/mst029

[30] A. Kouranov, L. Xie, J. de la Cruz, L. Chen, J. Westbrook, P. E. Bourne, and H. M. Berman, "The RCSB PDB information portal for structural genomics." *Nucleic acids research*, vol. 34, no. Database issue, pp. D302–5, Jan. 2006. [Online]. Available: http://nar.oxfordjournals.org/cgi/content/abstract/34/suppl_1/D302

[31] B. Seebeck, I. Reulecke, A. Kämper, and M. Rarey, "Modeling of metal interaction geometries for protein-ligand docking," *Proteins: Structure, Function, and Bioinformatics*, vol. 71, no. 3, pp. 1237–1254, Nov. 2007. [Online]. Available: http://doi.wiley.com/10.1002/prot.21818

[32] H. Zheng, M. Chruszcz, P. Lasota, L. Lebioda, and W. Minor, "Data mining of metal ion environments present in protein structures." *Journal of inorganic biochemistry*, vol. 102, no. 9, pp. 1765–76, Sep. 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016201340800127X

Pieter Meysman received the BS, MS and PhD degree in bioscience engineering from the KU Leuven, Belgium. He is currently a post-doctoral researcher at the Advanced Database Research and Modeling (ADReM) research group of the University of Antwerp and the Biomedical Informatics Research Centre Antwerp of the University of Antwerp and the University Hospital of Antwerp. He is currently the chair of the Student Council of the International Society for Computational Biology (ISCB-SC) since being elected in 2015. His research interests are mainly focused on data mining algorithms for pattern discovery in structural genomics, evolutionary transcriptomics and host-pathogen interactions.

Yvan Saeys is a professor at Ghent University and a group leader at the Flemish Institute for Biotechnology (VIB). He is heading the Data Mining and Modelling for Biomedicine (DaMBi) research group, an interdisciplinary team consisting of mathematicians, computer scientists, engineers and bioinformaticians. The main focus of the group is the design and application of novel data mining and machine learning techniques for applications in biomedicine and systems immunology. He has developed several state-of-the-art machine learning techniques that have been nominated as best performing techniques in the field of systems biology (winner of the DREAM5 and FlowCAP-IV challenges). He is an associate editor of the Information Fusion journal and has served on the program committee of numerous workshops focusing on the interplay between data mining and machine learning and bioinformatics.

Ehsan Sabaghian received his BS in statistics and MS degree in Biostatistics from Mashhad univerity of medical sciences, Iran. He currenlty is a PhD student in bioinformatics in group of Yves Van de Peer at Ghent university department of Plant Systems Biology which is a part of VIB. His main research is on describing whole genome duplication events in plants by looking on gene expression pattern of duplicated genes.

Wout Bittremieux received his BS and MS degree in computer science at the University of Hasselt. He currently is a PhD student in bioinformatics at the Advanced Database Research and Modelling (ADReM) research group of the University of Antwerp and the Biomedical Informatics Research Centre Antwerp of the University of Antwerp and the University Hospital of Antwerp. His research mainly focuses on data mining and machine learning techniques in the field of mass spectrometry-based proteomics.

Yves Van de Peer is professor in Bioinformatics and Genome Biology at Ghent University, Belgium; adjunct professor at the University of Western, London, Ontario (CA), at the Departments of Biology and Computer Science; and part time professor at the University of Pretoria, at the Genomics Research Institute.

Bart Goethals is professor at the Department of Mathematics and Computer Science of the University of Antwerp in Belgium. He leads the Advanced Database Research and Modeling (ADReM) research group, which performs fundamental research on the structures, the basic properties and the power of languages, algorithms and methodologies for processing and analysing large quantities of data. His primary research interests are the study of data mining techniques to efficiently find interesting patterns and properties in large databases. He received the IEEE ICDM 2001 Best Paper Award and the PKDD 2002 Best Paper Award for his theoretical studies on frequent itemset mining. He was organizer and program chair of ECML PKDD 2008, program chair of SIAM DM 2010, and general chair of IEEE ICDM 2012. He organised and chaired several workshops, such as ICDM FIMI 2003, 2004, PKDD KDID 2004, SIGKDD OSDM 2005, SDM HPDM 2006, and SIGKDD UP 2010 and served on the organizing and program committees of several conferences such as ACM SIGKDD, IEEE ICDM, SIAM DM, and ECML PKDD. He has served as general chair of the ECML PKDD Steering Committee, action editor of the Data Mining and Knowledge Discovery journal, and associate editor of the IEEE Transactions of Knowledge and Data Engineering, the Knowledge and Information Systems journal and was Editor-in-Chief of the ACM SIGKDD Explorations newsletter.

Kris Laukens is associate professor bioinformatics / biodata mining at the department of Mathematics & Computer Sciences at the University of Antwerp. He leads a bioinformatics group within the Advanced database Research and Modelling lab (ADReM). His scientific interest consists of data mining approaches to model and elucidate biological systems. He has a background in Biology, and finished a PhD in 2003 on the development and application of proteome analysis techniques. After his PhD, he worked on bioinformatics research projects within artificial intelligence and data mining research labs. He is founder and coordinator of the interdisciplinary research consortium biomina (biomedical informatics research network Antwerpen), jointly established by University of Antwerpen and University Hospital Antwerpen (UZA).