Graph Mining based clustering of PPI Networks

Abhinav Garlapati
CS11B030

Pallavi Gudipati
CS11B044

***Abstract*—An organism's protein-protein interaction network can convey key information about it. Here we propose a method to cluster organisms based on the structure of their protein-protein networks. We use an algorithm called GraphSig to mine significant subgraphs of the network database and cluster the organisms based on presence of these significant subgraphs in their networks. GraphSig is more scalable than its counterparts in significant subgraph mining domain and can mine significant subgraphs even when they are present in low frequencies. We evaluate our method based on our prior beliefs about the similarity between organisms. This method will lead to simplification in the process of analyzing a new or lesser known organism.**

Keywords*—*Graph Mining, PPI networks, GraphSig, Clustering, Similarity measures

I. Introduction

Over the years, researchers have collected and documented different types of protein-protein interactions in different organisms. Studying each network individually is not a feasible approach because of the size and complexity of the networks and the number of living organisms. In this project we try an alternative approach, which need not be new, but seems to be a reasonable method of analysis.

The inspiration behind this approach is a field of Computer Science called Graph Mining. Graph Mining is the process of finding useful information from large graphs. We have access to protein-protein interaction networks of different organisms, so we will be using this as our base data set, and mine significant subgraphs. Once we have significant subgraphs from all the organisms, we try to analyze the similarities of the networks among different organisms.

The final expected output of the project would be clusters of organisms which have similar protein-protein interaction networks. This result will be used to evaluate our approach by comparing it with our prior belief about the similarity among organisms (may be functional or evolutionary). The success of this approach will simplify the analysis of any new PPI network, as we will be able to group it to a known cluster and have some information about its basic behavior.

A. Objective

To implement a feasible technique to mine significant subgraphs from a graph database on a database of PPI networks and to use the results to cluster the networks.

B. Why this method?

Most of the existing algorithms use a naive frequent subgraph mining approach to get significant subgraphs. In this approach, all the subgraphs with frequency above a support threshold are obtained. Significance of each subgraph is obtained using a significance measure and the subgraphs that meet a significance threshold are returned. The problem arises when we try to choose a value for support threshold. If the threshold is too high, then we might miss significant subgraphs which have low frequency as frequency is not a measure of significance. If the threshold is too low but the running time of the algorithm grows exponentially with decreasing threshold due to the inevitable explosion of search space. GraphSig[1] is scalable technique that mines significant subgraphs even if the subgraphs have a low frequency.

We will give brief background in section III on certain topics. The method developed will be explained further in section IV. Assumptions of this model will be enumerated in section V. This will be followed by results obtained in section VII and discussion on the results obtained and other observations in section VIII. Section IX will conclude this project report. Also our ideas on future work possible will be summarized in section X.

II. Dataset

We are using the STRING database to get all the PPI networks. Each protein is considered to be a node and interaction of any type between two proteins is considered to be an edge. The weight of an edge is the combined score given in the database.

III. Background

A. Random Walk with Restarts

Random Walk simulates the trajectory taken by a random walker. At each node, the walker can move along the edges originating from that node with equal probability. As the walker goes along the path, we collect edge-types and node-types as features. We have used only the edge-types as features but the node-types can probably be used too. To limit the neighbourhood in which the random walker can walk, we add a restart probability, β. Thus at every node, with certain probability the walker can be teleported back to the intial node.

IV. Method

Our method can be broadly divided into the following phases:

- Phase 1: Pre-processing
- Phase 2: GraphSig
- Phase 3: Clustering

Fig.1 outlines our approach.

Definition 1. Frequent Subgraph. *Given a graph dataset $D = \{G_1, \cdots, G_n\}$ and a frequency threshold θ, a subgraph*

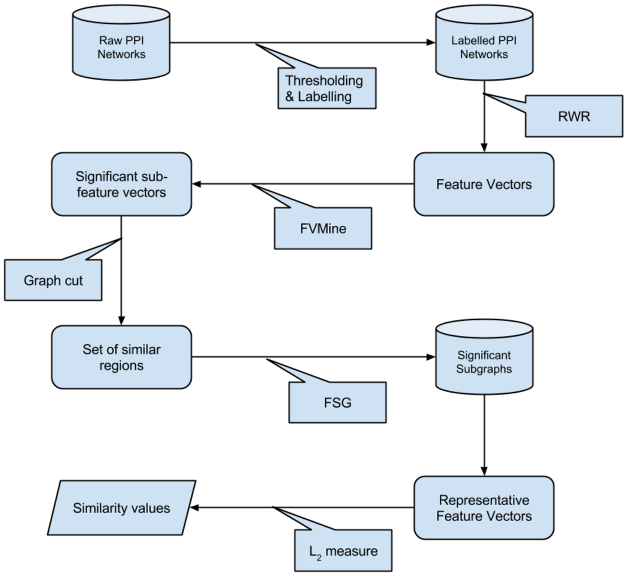

Fig. 1. Approach

g with an observed support μ_0 is frequent if and only if $\mu_0 \geq \frac{\theta|D|}{100}$.

Definition 2. Maximal Subgraph. *A subgaph g is a maximal subgraph in a graph dataset D if g is frequent, and there exists no supergraph $g^{'}$ such that $g \subset g^{'}$ and $g^{'}$ is frequent in D.*

Definition 3. Statistical Significance. *The statistical significance (or $p - value$) of a subgraph g with an observed support μ_0 is defined as the probability that it occurs in a random database with a support , where $\mu \geq \mu_0$.*

A. Phase 1.1: Converting the weighted graph into an unweighted graph

Each graph g_w in D_w is converted into an unweighted graph and put into D. This is done by applying a threshold on each edge and removing it if its weight is below the specified threshold. The weights of all the surviving edges is set to 1. Fig.2 and Fig.3 show E.coli's degree distribution before thresholding and after thresholding respectively.

B. Phase 1.2: Making node and edge types

For each graph g in D, the nodes are sorted based on their degree. They are then uniformly put into bins. Each bin represents a node type and all the nodes in it are labelled with the bin number as its node type. Edge type is an unordered tuple of the node types of the vertices corresponding to the edge. Our intuition behind this is that degree is a measure of importance. Thus our nodes are classified on the basis of their importance

C. Phase 2.1: Graph to feature vectors

Each graph g in D is converted into a set of feature vectors. This is achieved by performing Random Walk with Restarts on the graph by taking a different node as the point of focus every time till all the nodes have been considered. RWR is run till the feature vector converges. All these feature vectors are put in a single set F.

D. Phase 2.2: Partitioning the feature vector set

The feature vector set F is partitioned such that all the feature vectors generated by taking the node labelled α as the point of focus are in the set F_α. Thus, all the feature vectors are grouped based on the importance of their generator nodes.

E. Phase 2.3: FVMine[2]

Definition 4. Closed vector. *A feature vector x is closed if none of its super-feature vectors has the same support as x.*

Definition 5. Floor of a Vector. *The floor of a set of vectors $\{v_1, \cdots, v_n\}$ is a vector v_f where, $v_{f_i} = min(v_{1_i}, \cdots, v_{n_i})$ for i = 1 $\cdots$ n. Ceiling of a set of vectors is defined analogously.*

Feature Vector Mining(FVMine) is performed on each subset F_α and a set of significant sub-feature vectors are extracted. The p-value threshold is chosen empirically and minimum support threshold is kept low so that less frequent significant feature vectors are not ignored. All these vectors are put in the set S_α.

```
Algorithm 1 FVMine($x, S, b$)
Require: $x$ is current sub-feature vector
Require: $S$ supporting set of $x$
Require: $b$ current starting position
Ensure: $A$ is the set of all significant sub-feature vectors
if $p - value(x) \leq maxPvalue$ then
    $A \leftarrow A + x$
end if
for $i = b$ to $m$ do
    $S^{'} \leftarrow \{y | y \epsilon S, y_i > x_i\}$
    if $|S^{'}| < minSup$ then
        continue
    end if
    $x^{'} = floor(S^{'})$
    if $\exists j < i$ such that $x^{'}_j > x_j$ then
        continue
    end if
    if $p - value(ceiling(S^{'}, |S^{'}|) \geq maxPvalue$ then
        continue
    end if
    $FVMine(x^{'}, S^{'}, i)$
end for
```

F. Phase 2.4: Extracting regions of interest

For each vector v, in S_α, regions around the nodes that are labelled α and their corresponding feature vector is a super-vector of of v are regions of interest. The whole database of graphs, D, is scanned for such regions of interest. After such regions are located, a subgraph with an empirically determined radius is isolated which is centered at the node labelled α. All these sub-graphs are put in the set, G_v.

G. Phase 2.5: Maximal subgraph mining

Maximal subgraph mining is performed on G_v with a high threshold as we expect all the graphs in G_v to have a common subgraph. We finally get a set of maximal subgraphs which are put in the set M. We haven't written our own code to mine maximal subgraphs and have used PAFI Software Package. The mining technique used by PAFI is highly optimized.

Algorithm 2 GraphSig($G, minSup, maxPvalue$)

```
Require: $G$ is a graph database
Require: $minSup$ is the support threshold
Require: $maxPvalue$ is the p-value threshold
Ensure: $A$ is the answer set of all significant subgraphs
D ← ∅
A ← ∅
for each $g \epsilon G$ do
    $D \leftarrow D + RWR(g)$
end for
for each node-type $\alpha$ in $G$ do
    $D_\alpha \leftarrow \{v | v \epsilon D, label(v) = \alpha\}$
    $S \leftarrow FVMine(floor(D_\alpha), D_\alpha, 1)$
    for each vector $v \epsilon S$ do
        $N \leftarrow \{n | nisanodeoflabel\alpha, v \subseteq vector(n)\}$
        $E \leftarrow \emptyset$
        for each node $n \epsilon N$ do
            $E \leftarrow E + CutGraph(n, radius)$
        end for
        $A \leftarrow A + MaximalFSM(E, freq)$
    end for
end for
```

H. Phase 3.1: Graph to feature vector

Each graph g in D is converted to a feature vector, f. The length of the feature vector will be the size of the set of maximal subgraphs extracted in Phase 2.5, $|M|$.

$$f_i = frequency(M_i, g). \tag{1}$$

I. Phase 3.2: Comparison of feature vectors

Once we get all the graphs into feature vector space, we can use any existing clustering technique. Since our dataset has only few organisms, it doesnt make sense to use traditional clustering. We have just compared the networks by using different measures to calculate the distance between their corresponding feature vectors.

V. Assumptions

A. Model Adaption

- Degree of a node is a measure of its importance.
- The similarity between two graphs can be modelled using just the Euclidean distance between their representive vectors that are generated.

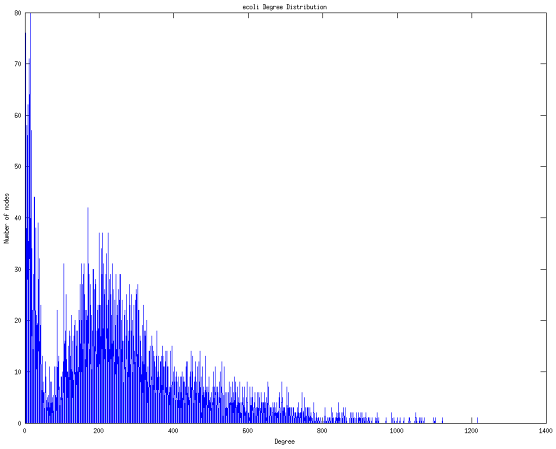

Fig. 2. E.coli degree distribution

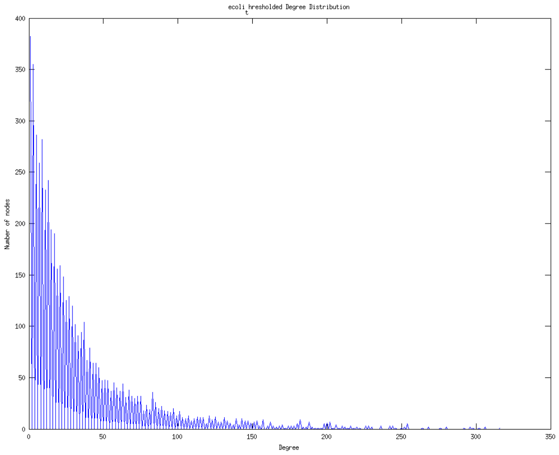

Fig. 3. E.coli degree distribution after thresholding

B. GraphSig

- Independence of the features in the calculation of empirical probability.
- Regions of interest have a common subgraph if the floor of the set of corresponding feature vectors has non-zero values.
- Low p-value in the feature space corresponds to low p-value in graph space.

VI. Procedure

A. Parameter values used

Table I gives all the parameter values used.

B. Networks considered

- **Escherichia coli O6:K15:H31 536**
- **Escherichia coli O103:H2 12009**
- **Escherichia coli C ATCC 8739**

TABLE I. PARAMETER VALUES

Parameter	Use	Value
$edgeThresh$	Threshold for unweighted graph conversion	600
β	Restart probability in RWR	0.067
$minSup$	Mininmum support in FVMine	0.1%
$maxPvalue$	Maximum p-value in FVMine	0.01
$cutoffradius$	Radius used in GraphCut	2
$fsgSup$	Minimum support used in FSG	80%

TABLE II. E.COLI STRAINS

Strain	Phylo group	Pathogenecity
Escherichia coli O6:K15:H31 536	Group B2	UPEC
Escherichia coli O103:H2 12009	Group B1	EHEC
Escherichia coli C ATCC 8739	Group A	-
Escherichia coli O127:H6 E2348/69	Group B2	EPEC

TABLE III. EUCLIDEAN DISTANCES

Organism-1	Organism-2	Distance
E.coli O6:K15:H31 536	E.coli O103:H2 12009	66.7233092704
E.coli O6:K15:H31 536	E.coli C ATCC 8739	74.9466476902
E.coli O6:K15:H31 536	E.coli O127:H6 E2348/69	84.7761758987
E.coli O103:H2 12009	E.coli C ATCC 8739	22.2485954613
E.coli O103:H2 12009	E.coli O127:H6 E2348/69	45.3982378513
E.coli C ATCC 8739	E.coli O127:H6 E2348/69	48.3735464898

- **Escherichia coli O127:H6 E2348/69**

Table II gives some information about the selected strains.

VII. RESULTS

A. Distances measured

Table III gives the distances computed.
Some other results obtained:

- Overall 34 maximal frequent subgraphs were obtained.
- The biggest subgraph obtained(node wise) had 7 nodes and 6 edges.
- Fig.4 and Fig.5 show some of the maximal frequent subgraphs obtained.

VIII. DISCUSSION

A. Why not other organisms?

The reason we took only strains of E.coli was that the networks corresponding to them were similar to a large extent. We observed that if the networks are too far apart, then $FVMine$ takes a unreasonably long time to run. Also, sometimes the results are very skewed, as in the significant subgraphs from one organism totally overshadows the subgraphs from other organisms. This caused some of the organisms to not have any representation in the final output of significant graphs.
Another interesting observation was that for highly significant nodes, almost the whole graph was reachable within 5 steps. Thus even if the radius in graph-cut is around 3 or 4, we were getting a significant portion of the graph.
We also observed that the strains that are farthest apart also belong to the same phylogenetic group. We can infer that the network structure, atleast the way we are interpreting it, doesn't correspond to phylogeny. It may correspond to the functionality of the strain, or if applicable its pathogenity, but its difficult to say that with only four data points.

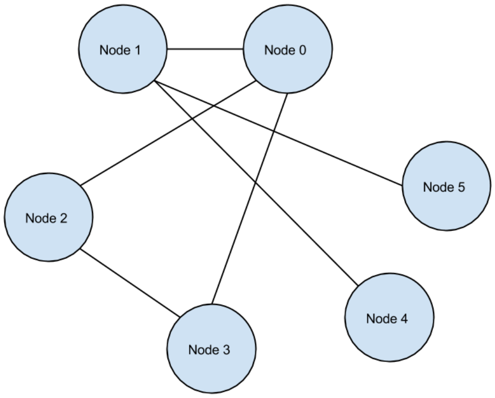

Fig. 4. Maximal FS: 6 nodes, 6 edges

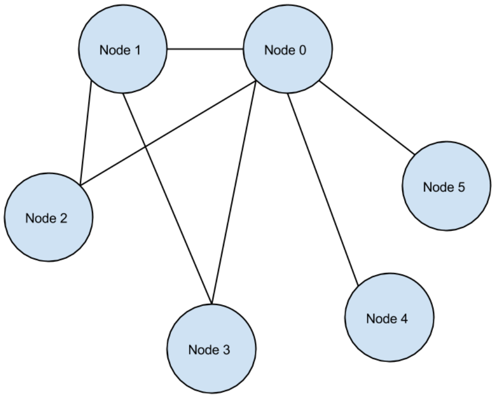

Fig. 5. Maximal FS: 6 nodes, 7 edges

IX. CONCLUSIONS

We find that the algorithm GraphSig is applicable to PPI networks too. Also it is scalable to a certain extent. We also needed to adapt it to PPI networks. We have not experimented a lot when the graphs are too dissimilar but the algorithm might need further tweaking and regularization. Also, the scalability of the algorithm is doubtful in such cases.
We can also conclude that the phylogeny of the organism might not affect the structure of the network, atleast in the way we are interpreting the networks.

X. FUTURE WORK

Our first step would be to attach more biological sigificance to the results. More in-depth study of the organisms chosen would facilitate this. Another possible direction would be to adapt this method to weighted graphs. When we convert

weighted graphs to un-weighted graphs, we can loose significant informations that could lead to detrimental results.
To counter the skew generated in $FVMine$ when very dissimilar organisms are used, we could introduce a regularizer to limit the number of significant graphs contributed by one organism.

A. Transition from Graph metrics to biological metrics

The labelling of the nodes was done on the basis of its degree. We used degree as a proxy for the significance of a protein. A possible improvement would be to get the significance of a protein from literature and use that to label the nodes. This would increase the extent of domain specific knowledge used and thus tune the model to PPI networks.
In our model, we take edge types as the set of features. The type of an edge is decided by the significance of the vertices that the edge connects. To improve the model, we can take the edge type to the type of interaction between the proteins rather than the significance of the proteins.

References

[1] S. Ranu and a. K. Singh, GraphSig: A Scalable Approach to Mining Significant Subgraphs in Large Graph Databases, *2009 IEEE 25th Int. Conf. Data Eng.*, pp. 844855, Mar. 2009.

[2] H. He and A. Singh, GraphRank: Statistical Modeling and Mining of Significant Subgraphs in the Feature Space, *Sixth Int. Conf. Data Min.*, pp. 885890, Dec. 2006.

[3] L. T. Thomas, S. R. Valluri, and K. Karlapalem, Margin: Maximal Frequent Subgraph Mining, *ACM Trans. Knowl. Discov. Data*, vol. 4, no. 3, pp. 142, Oct. 2010.

[4] X. Yan, X. Yan, H. Cheng, H. Cheng, J. Han, J. Han, P. Yu, and P. S. Yu, Mining significant graph patterns by leap search, in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 433444.

[5] R. R. Chaudhuri and I. R. Henderson, The evolution of the Escherichia coli phylogeny., *Infect. Genet. Evol.*, vol. 12, no. 2, pp. 21426, Mar. 2012.