GraphClust: a Method for Clustering Database of Graphs

D. Reforgiato Recupero[1], D. Shasha[2]

[1]Dipartimento di Matematica e Informatica

Università degli Studi di Catania

e-mail: diegoref@dmi.unict.it

[2]Computer Science Department

New York University

e-mail: shasha@cs.nyu.edu

Abstract

Any application that represents data as graphs may be interested in finding patterns in those graphs. To do this in an unsupervised fashion requires the ability to find subgraphs that are similar to one another. That is the purpose of GraphClust. GraphClust is an algorithm and software that clusters directed and undirected labelled graphs. The algorithm proceeds in three phases: it finds (first phase) highly connected substructures in each graph and then (second phase) it uses those substructures to represent each graph as a feature vector. Clustering (third phase) itself can be done using the k-means or the Antipole method, though other methods are of course possible. We validate the cluster quality by using the silhouette method. Moreover, SVD decomposition leads to the computation of highly co-occurring substructures.

Index terms - Text clustering, Document vectors, Graphs clustering, Graphs substructure.

1 Related Work

In the last few years, developing algorithms for clustering data represented by graphs has been recognized as a problem in the pattern recognition community [10]. Nevertheless, graph clustering is still an open problem for two reasons. First, many interesting exact graph matching problems, e.g. subgraph isomorphism, maximum common subgraph, etc., are NP-complete and then exact graph clustering algorithms using graph matching are extremely time consuming. Second, the proper distance metric between graphs is a matter of debate.

In chemistry, the clustering of substances is of central interest [7]. Starting from molecules of known properties, compounds that are structurally similar are likely to exhibit similar properties [27]. Moreover, for having a successful clustering, an appropriate description of molecule structures and an adequate clustering algorithm are both essential. Methods based on the relevant compound property space has been applied for the prediction of physiochemical and biological properties of chemical compounds [8, 9], the selection of diverse representative compounds and reagent sets for the design of combinatorial libraries [22], compound acquisition selection [37, 32] and the compilation of screening sets [16]. Many methods mostly based on fingerprint descriptions have appeared in literature [18]. The hierarchical Ward [45], the nonhierarchical Jarvis-Patrick [26] and the k-means relocation methods [15] are the most popular. Ward's method outperforms the other two in chemical databases [7, 16, 18, 46]. Ott et al. [34] proposed a sequential superparamagnetic clustering approach which clusters correctly datasets composed of structures from seven chemically distinct compound classes.

There is an extensive literature on subgraph searching [40, 39]. Most of these methods are designed for specific applications. For example, Daylight [25], proposed a searching system for molecular databases using fingerprints consisting of bit vectors, where each position is associated to a small path. It outputs all the molecules that contain at least one occurrence of the query.

Other methods for XML databases have been proposed [12, 31, 38, 19, 36].

Messmer and Bunke [33] proposed a method which indexes the graphs in a database and computes a graph isomorphism. Both indexing and matching are based on all possible permutations of the adjacent matrices of the graphs.

GraphGrep [20] is a method which finds all the occurrences of a subgraph in a database of graphs. Its filtering process starts by computing for each graph and for each node all the paths starting at that node and having length one up to a small constant. An hash-table associates all the paths with their corresponding indexes.

Text retrieval [30, 13, 3, 29] has focussed on the need to locate textual information efficiently. A classical text searching method [13] involves modelling a text collection in document-term matrix, and evaluating a document's relevance to a query using a linear algebraic dot product. In a term-document matrix A, $A[i, j]$ gives the number of occurrences of term j in document i. Queries are normally represented as a bit vector over the same set of terms. The similarity between document vectors (the rows of document-term matrices) can be found by their inner product. This corresponds to determining the number of term matches (weighted by frequency) in the respective documents. Another commonly used similarity measure is the cosine of the angle between the document vectors. This can be achieved computationally by first normalizing (to 1) the rows of the document-term matrices before computing inner products. Singular Value Decomposition (SVD) has been shown to work well for text retrieval over the last fifteen years [14, 23]. The motivation is simple: large document-by-term matrices have a significant amount of redundant data. Removing this information allows a more precise and efficient search. Singular Value Decomposition achieves rank reduction by breaking the matrix A in the product of 3 matrices T, S, D^T which are truncated to r dimensions.

Latent Semantic Indexing (LSI), [24], attempts to project term and document vectors into a lower dimensional space spanned by the true "factors" of the collection. This uses a truncated Singular Value Decomposition (SVD) of the term-document matrix.

Spectral methods try to represent the most interesting properties of the input graphs using vectors, thus reducing the graph clustering problem to a problem in a vector space [4, 5, 28]. This allows a new *spectral* method, closely related to latent semantic indexing, to be used.

Subdue is another method to capture essential structure information from graphs. The Subdue substructure discovery system, [2], discovers repetitive subgraphs in a labelled graph representation by using the minimum description length principle. Experiments show Subdue's applicability to several domains, such as molecular biology, image analysis and computer-aided design.

In this paper, GraphClust, a new algorithm for clustering labelled graphs, will be presented. The problem of mapping the graphs as feature vectors is solved by creating some substructures such that the frequency of substructure j in the graph i is stored at $A[i, j]$. After this, the rows of the matrix A are finally clustered. According to the kind of graph, GraphClust has two different methods for finding substructures: one introduced in GraphGrep [20] for the paths generation, more effective when the input graphs have many labels; the other technique is represented by the SUBDUE method [2], which is more suitable in presence of sparse graphs having few labels compared to the number of nodes, like e.g. chemical compounds. A list of highly correlated substructures is at the end created by using the SVD reduction.

From Diego to DENNIS: the following paragraph is new. From Dennis to Diego: edited, but still the question is: Algorithmically what are we doing that is different? From Diego to Dennis: here as our innovation I want to claim our trade-off between scalability and performance since the previous methods are focussed on small datasets of chemical compounds. From Diego to Dennis: I mention the performance section where we have a time-complexity graph on NCI database

In contrast to previous work ([27, 8, 9, 34]), where particular properties of the domain are taken into account (e.g. about the similarity of chemical structures), GraphClust works with any kind of labelled graph. Clustering chemical compounds by using the Jarvis-Patrick [26] or Daylight System [25] or the Ward's method [45] could be computationally prohibitive for large databases. The AllPairShortestPath procedure to find important substructures and either k-means or the Antipole method for clustering the resulting feature vectors according to the underlined similarity measure makes GraphClust a fast and effective method able to perform well also on large datasets as shown in section 5.

2 Design

GraphClust assumes that each node of the database graphs has a unique identification number and a label. Edges are unlabelled (for purpose of this paper).

GraphClust deals with either directed or undirected graphs. The substructures can

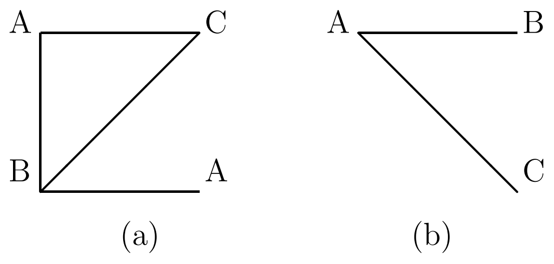

Figure 1: Dataset of two graphs.

Graph (a)		Graph (b)	
Initial Node	Substructures generated	Initial Node	Substructures generated
top-left A	{A,AC,AB,ABA}	A	{A,AB,AC}
B	{B,BC,BA,BA}	B	{B,BA,BAC}
bottom-right A	{A,AB,ABC,ABA}	C	{C,CA,CAB}
C	{C,CA,CB,CBA}		

Table 1: Patterns generated from the dataset of Fig. 1 using AllPairShortestPath with $l_p = 3$.

	C	CA	CB	CBA	A	AB	ABA	B	BAC
graph (a)	1	2	2	2	2	4	2	1	0
graph (b)	1	2	0	0	1	2	0	1	2

Table 2: Matrix A generated from the patterns of Table 1.

be discovered in two ways:

- by using the AllPairShortestPath algorithm as shown in [20]; in this case, for each graph of the dataset and for each vertex v, all shortest paths whose lengths are at most l_p are generated from v. Each path is represented by the sequence of node labels in that path.
- by using the Subdue substructure discovery system ([2]); in this case, for each graph g of the dataset, Subdue finds common or approximately common substructures of g.

These two techniques to find substructures work well regardless of the source application domain of the graph as we show in our experimental section.

A matrix having a number of columns equal to the number of found substructures and a number of rows equal to the number of the graphs in the dataset is created. Each entry $A[i, j]$ represents the number of times in which the substructure j is contained in the graph i.

If AllPairShortestPath is run, it finds for each graph all the label sequences corresponding to all paths whose lengths are at most l_p and therefore it creates more columns in the matrix A than Subdue. Experimentally we observed that $l_p = 4$ is a good tradeoff between time and precision. However, if too many substructures are found with either AllPairShortestPath or Subdue, GraphClust considers only the max_sub most frequent, where max_sub is a constant of the system. For example, on chemical compounds, where usually there are not so many nodes and edges, Subdue provides a better solution in terms of clustering precision. For graphs with many edges, AllPairShortestPath outperforms Subdue.

Once that the matrix A is completed, we cluster its rows. There are two possible clustering algorithms to use: one is the k-means algorithm in which the user chooses the number of clusters k to create; the other is the Antipole clustering [11] in which the user chooses a "tightness" measure (an integer value in the range 1 to 4) where the higher the measure the smaller the cluster radius and hence the larger the number of generated clusters. Antipole clustering [11] is much faster than k-means even if it is not possible to know a-priori the number of clusters that will be created. The metric distance used in both clustering algorithms just described can be either Euclidean distance or inner product distance. Euclidean distance is appealing for applications having a natural geometry. Inner product is better for non-spatial applications such as text-similarity.

The above clustering algorithms work well across all the domains we have tried.

In table 2 a matrix obtained from the patterns of table 1 generated by applying the AllPairShortestPath algorithm with $l_p = 3$ to the dataset in Fig. 1 is shown.

Another operation we perform when the matrix A is complete, is the creation of correlated substructures. By using the SVD method, the substructure-graph matrix A^T is broken apart into the product of 3 matrices T, S and D^T. Table 3 shows a SVD of the matrix A in table 2. These matrices are truncated to r dimensions with r chosen by the user. Dimensionality reduction reduces the noise present in the substructure-substructure matrix revealing a more robust relationship between the substructures. The substructure-substructure correlation matrix X_r is then computed by multiplying $T_r \times S_r \times (T_r \times S_r)^T$. Table 4 shows the substructure-substructure correlation matrix X_r computed with the matrices T_r, S_r of table 3 reduced for $r = 1$. When the number of substructures is too large and then it would be too expensive to compute the singular value decomposition, GraphClust considers only the main substructures (the ones with highest support).

$$\begin{pmatrix} -0.20 & -0.17 \\ -0.40 & -0.33 \\ -0.26 & 0.35 \\ -0.26 & 0.35 \\ -0.33 & 0.01 \\ -0.66 & 0.02 \\ -0.26 & 0.35 \\ -0.20 & -0.17 \\ -0.13 & -0.68 \end{pmatrix} \times \begin{pmatrix} 6.8 & 0 \\ 0 & 2.61 \end{pmatrix} \times \begin{pmatrix} -0.89 & 0.46 \\ -0.46 & -0.89 \end{pmatrix}$$

$$T \quad \times \quad S \quad \times \quad D^T$$

Table 3: Matrices T, S, D^T generated by the Singular Value Decomposition of A^T of Table 2.

3 Algorithms

It turns out that GraphClust consists of 16 different algorithms broken down along the four binary dimensions described in the section 2. The main concept of GraphClust is the mapping of the data graphs into k-dimensional vectors. To perform this step

$$\begin{pmatrix} -0.20 \\ -0.40 \\ -0.26 \\ -0.26 \\ -0.33 \\ -0.66 \\ -0.26 \\ -0.20 \\ -0.13 \end{pmatrix} \times \begin{pmatrix} 6.8 & 0 \end{pmatrix} \times \left(\begin{pmatrix} -0.20 \\ -0.40 \\ -0.26 \\ -0.26 \\ -0.33 \\ -0.66 \\ -0.26 \\ -0.20 \\ -0.13 \end{pmatrix} \times \begin{pmatrix} 6.8 & 0 \end{pmatrix} \right)^T =$$

	C	CA	CB	CBA	A	AB	ABA	B	BAC
C	2	4	2	2	3	6	2	2	2
CA	4	8	4	4	6	12	4	4	4
CB	2	4	4	4	4	8	4	2	0
CBA	2	4	4	4	4	8	4	2	0
A	3	6	4	4	5	10	4	3	2
AB	6	12	8	8	10	20	8	6	4
ABA	2	4	4	4	4	8	4	2	0
B	2	4	2	2	3	6	2	2	2
BAC	2	4	0	0	2	4	0	2	4

Table 4: Reduced correlation substructure-substructure matrix $X_r = T_r \times S_r \times (T_r \times S_r)^T$ for $r = 1$.

we have introduced the concept of substructures and the methods used to find these substructures.

In this section, the algorithms used by GraphClust in the three steps of its main procedure will be discussed.

Subdue [2] discovers interesting and repetitive subgraphs in a labelled graph representation using the minimum description length principle; Subdue discovers substructures that compress the original data and represent structural concepts in the data. By replacing previously-discovered substructures in the data, multiple passes of Subdue produce a hierarchical description of the structural regularities in the data. Subdue uses a computationally-bounded inexact graph match that identifies similar, but not identical, instances of a substructure and finds an approximate measure of closeness of two substructures when under computational constraints. In addition to the minimum description length principle, other background knowledge can be used by Subdue to guide the search towards more appropriate substructures. Once the substructures and the matrix A have been created, the clustering is performed by the k-means or Antipole clustering method [11]. In our implementation of k-means, at the first iteration (that is for $t = 1$), the initial k centroids $q_1^1, q_2^1, \ldots, q_k^1$ are computed by using the Gonzalez [21] algorithm; then, the rest of the objects are assigned to a class according to the relation $x_l \in C_j^1$ iff $d(x_l, q_j^1) \leq d(x_l, q_i^1)$, $1 \leq j,\ i \leq k,\ i \neq j$. At the generic iteration t, new centroids are computed in such a way that the performance index, $\gamma_i = \sum_{x \in C_i^t} |x - q_i^t|^2, i = 1, 2, 3, \ldots, k$, is minimized. This is achieved making $q_i^{t+1} = \frac{1}{n_i^t} \sum_{\forall x \in C_i^t} x$. If $q_i^{t+1} = q_i^t$, the process finishes, otherwise, the objects are grouped again.

The Antipole clustering [11] algorithm of bounded radius is performed by a top-down procedure starting from a given finite set of points S which checks if a given splitting condition is satisfied. This condition asks for two points whose distance is greater than the radius. If there are no two such points, then splitting is not performed and the given subset is a cluster on which an approximate centroid is then found. Otherwise, a suitable pair of points (A, B) of S called Antipole is generated and the set is partitioned by assigning each point of the splitting subset to the closest endpoint of the Antipole (A, B). As seen in [11] the randomized algorithms used by Antipole clustering makes its construction much faster than k-means's.

4 Complexity

Here is a description of the worst case complexity for the three steps of GraphClust. Let $|D|$ be the number of graphs in a database D. The first and second step of the algorithm depend on which algorithm is used to create the patterns. If AllPairShortestPath is used, then the complexity of the first step is $\mathcal{O}(\sum_i^{|D|}({n_i}^3))$, where n_i are the number of nodes of the graph i; in this case the complexity of the second step is $\mathcal{O}(|D||pat|\sum_i^{|D|}(n_i m_i^{l_p}))$, where $|pat|$ is the total number of patterns generated and m_i is the number of patterns starting from n_i. If the Subdue algorithm is used, then the complexity of the first step becomes $\mathcal{O}(\sum_i^{|D|}(\sum_{j=1}^{nsubs}(ninst_j \times gm_j)))$, where $ninst_i$ is the maximum possible number of non-overlapping instances for substructure j and gm_j is the user-defined maximum number of partial mappings that are considered during a graph match between substructure definition j and a potential instance of the substructure. In this case the complexity of the second step is $\mathcal{O}\ (|D||pat|\sum_i^{|D|}(\sum_{j=1}^{nsubs}(ninst_j \times gm_j)))$, where $ninst_i$ and gm_j have already been described above. Details of the Subdue complexity analysis can be found in [35].

For the third step we will consider for first the clustering process and then the

SVD computation. The clustering process complexity depends on which algorithm is used. K-means takes time $\mathcal{O}(tk|D|)$, with k the number of clusters and t is the number of iterations. Normally, $k, t << |D|$. The Antipole algorithm [11] has a worst-case complexity of $\frac{\tau(\tau-1)}{2}|D| + o(|D|)$ in the input size $|D|$, where τ is the bounded radius (see [11] for further details).

The speed of AllPairShortestPath makes it preferable to Subdue in presence of big datasets having large number of edges.

Finally, the SVD process complexity takes $\mathcal{O}\left(|pat|^2 \cdot |D| + |pat| \cdot |D|^2\right)$ so it would not be pratical for big datasets. To make the process effective also with large datasets, the SVD considers only the max_sub substructures more interesting (more repetitive in the database) where max_sub is a constant of the algorithm.

5 Performance Studies

We start with a systematic evaluation of the quality of the clusters. The Silhouette method [6] is used to show how good the clustering obtained is. Moreover, we want to show how well the final clustering performed by GraphClust captures the graphs present in different categories known *a-priori*. For this, we have used an artificial graph generator [17] to create a database containing five different categories of undirected graphs. The five categories includes randomly graphs, regular 2D-meshes, regular 3D-meshes, irregular 2D-meshes, irregular 3D-meshes. For each category we have generated 1000 graphs with 30 nodes and 1000 graphs with 80 nodes. The number of edges vary from 50 to 200. Each group of 1000 graphs differs for the 7% of the edges. Thus, our artificial dataset contains 10000 graphs. An optimal clustering creates 10 clusters, each one containing one structural group of graphs. Tables 5 and 6 depict the global silhouette values, GSu, for our obtained clustering whose number of clusters c varies from 10 to 15. Clustering in table 5 has been performed by using the Antipole Tree data structure tuning properly the radius such that the resulting clusters ranged in $[10, 15]$ whereas clustering in table 6 has been performed by using the k-means algorithm with $k \in [10, 15]$. For both clustering algorithms, the substructures have been discovered by using the AllPairShortestPath fixing the constant $l_p = 4$. In both tables $c = 10$ is suggested as the best clustering configuration for the examined data set and this is also the optimal number of clusters known for when the data set has been created.

c	GSu
10	0.712
11	0.665
12	0.693
13	0.607
14	0.518
15	0.489

Table 5: Global Silhouette values for clustering obtained by using GraphClust with Antipole Tree data structure.

c	GSu
10	0.886
11	0.830
12	0.714
13	0.678
14	0.643
15	0.660

Table 6: Global Silhouette values for clustering obtained by using GraphClust with k-means algorithm.

Now, we have to show that this clustering is also coherent with the *a-priori* classification of the data set. Recall that in the optimal clustering each cluster contains a single structural group where in each group the graphs differ by 7% of their edges. Table 7 shows the similarities in percent between the best clustering obtained in table 5 and 6 (that is for $c = 10$) and the optimal clustering. A value $x\%$ for the cluster C_i obtained with GraphClust means that C_i is equal to $x\%$ of the optimal cluster C_i. To measure the robustness of the clustering obtained, a pair of graphs g_1 and g_2 are considered to be consistent in the two clusterings if they are in the same cluster in both cases or in different clusters in both cases. Otherwise they are inconsistent. In table 8, for $c = 10$, the output clustering generated by GraphClust has been compared with the optimal clustering. The value *consistent_value* shows the number of graphs pairs that are consistent divided by the total number of graphs pairs for the output obtained.

From Diego to Dennis: I inserted a time-complexity graph on the same database NCI used by graphgrep in icpr2002

GraphClust performs well on large datasets. In Fig. 2 we show the time complexity (in seconds) of GraphClust on NCI databases up to 50000 molecules. We also report the number of substructures created for each dataset. The graphs in these databases

Clustering Algorithm	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
Antipole Tree	100%	100%	100%	100%	100%	100%	95.14%	66.0%	50%	28.9%
K-means	100%	100%	100%	100%	100%	100%	100%	54.4%	50%	45.6%

Table 7: Similarities in percentual between the best clustering found ($c = 10$) in Tables 5, 6 and the optimal clustering.

Clustering Algorithm	Number of consistent pairs	Total number of pairs	*consistent_value*
Antipole Tree	48704861	49995000	0.9741
K-means	48746936	49995000	0.9750

Table 8: Robustness between the best clustering found in Tables 5, 6 and the optimal clustering.

have an average number of 20 nodes; several graphs have up to 270 nodes. GraphClust has been executed by using AllPairShortestPath to find substructures and Antipole algorithm for clustering.

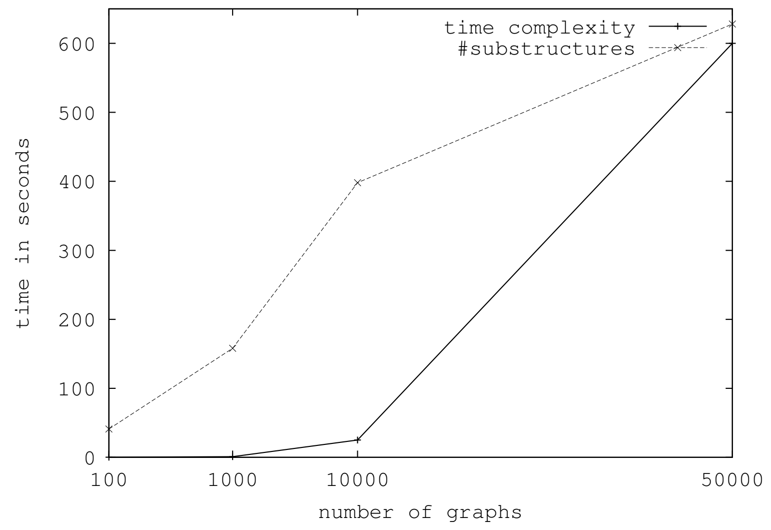

Figure 2: Time complexity and number of substructures for NCI database.

6 Finding gene co-regulations in biological data

In this section we analyze how GraphClust substructure discovery process is effective and present some results obtained on two biological datasets having to do with RNA expression. Recent microarray analysis have demonstrated that the expression of many regulatory genes are affected by nitrate [41, 44, 42]. In [43], microarray analysis of gene expression revealed that 595 genes responded to nitrate in both wild-type and mutant plants. We have used the outputs produced in [43] consisting of a list of correlated genes pairs that responded to nitrate in wild-type plants and in NR-null (nitrate reductase) plants. To study these two sets of correlation pairs at the functional level, we replaced each gene with the the Gene Ontology (GO) terms that characterize it. At the end we have obtained two Gene Ontology term graphs, one for the correlations of gene expression values in wild-type experiments and the other for the correlations of gene expression values in a (NR)-null double mutant of Arabidopsis. Each node of each graphs corresponds to a gene involved in some correlation and is labelled by the set of Gene Ontology terms associated with that gene. There is an edge between two nodes if there is a correlation between the expression of the two genes.

Given the size of the two resulting graphs, (13867 nodes and 999658 edges for the wild-types and 13428 nodes and 1022692 edges for (NR)-null double mutant), we have used AllPairShortestPath instead of Subdue for building paths of length up to $l_p = 4$.

Fig. 3 shows the GO-term paths resulting from administering nitrate on wild-type plants whereas Fig. 4 shows the results on mutant plants whose ability to assimilate nitrate is impaired. Our biological collaborator Rodrigo Gutierrez concludes from these figures that transcriptional genes are co-regulated with genes involved in other processes for the wildtype but not for the mutant. "One interpretation of this result is that the transcriptional response requires assimilation of nitrate into reduced forms of organic N sources."

From Diego to Dennis: I put this paragraph in a non-titled section and added the info of the NCI database since we are talking about the experiments time and the pc used

For all the experiments we used a Mobile Intel Pentium Processor 2.30GHz and 512MB of RAM with the Linux operating system. For the syntethic experiments the algorithm runs in about 10 minutes while for the real dataset it runs in about 20 hours. For the NCI database it takes much less since the average number of nodes of graphs is much smaller.

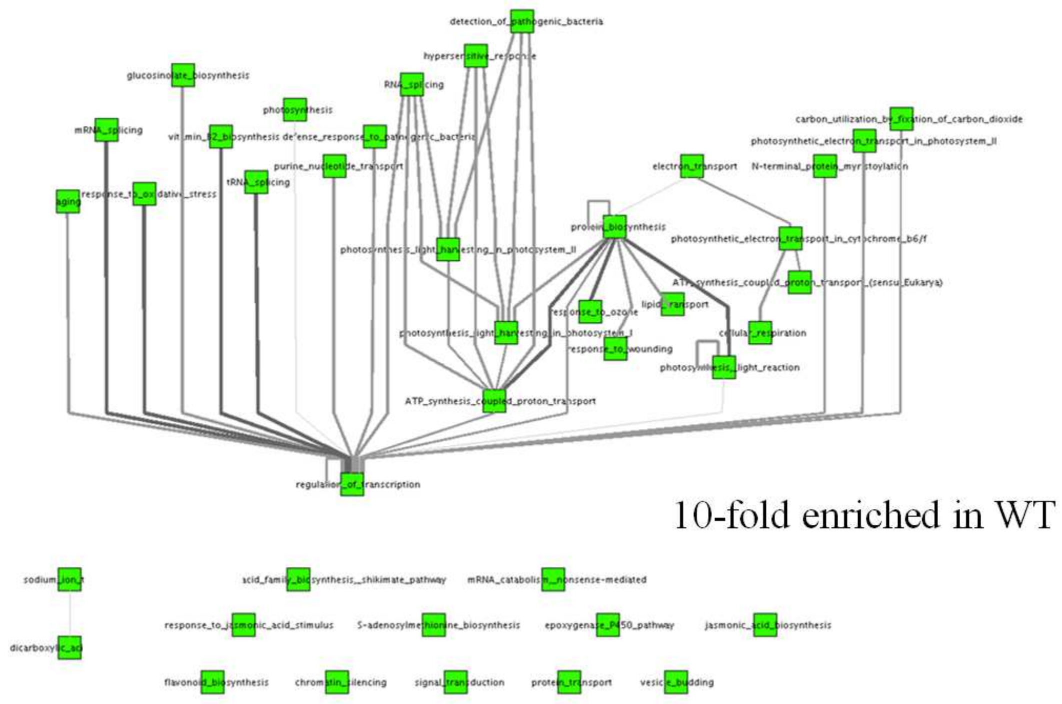
detection_of_pathogenic_bacteria
hypersensitive_response
glucosinolate_biosynthesis
RNA_splicing
photosynthesis
carbon_utilization_by_fixation_of_carbon_dioxide
mRNA_splicing
vitamin_B2_biosynthesis
defense_response_to_pathogenic_bacteria
photosynthetic_electron_transport_in_photosystem_II
purine_nucleotide_transport
electron_transport
N-terminal_protein_myristoylation
tRNA_splicing
aging
response_to_oxidative_stress
protein_biosynthesis
photosynthesis_light_harvesting_in_photosystem_II
photosynthetic_electron_transport_in_cytochrome_b6/f
ATP_synthesis_coupled_proton_transport_(sensu_Eukarya)
response_to_ozone
lipid_transport
photosynthesis_light_harvesting_in_photosystem_I
cellular_respiration
response_to_wounding
photosynthesis_light_reaction
ATP_synthesis_coupled_proton_transport
regulation_of_transcription
10-fold enriched in WT
sodium_ion_t
acid_family_biosynthesis_shikimate_pathway
mRNA_catabolism_nonsense-mediated
dicarboxylic_ac
response_to_jasmonic_acid_stimulus
S-adenosylmethionine_biosynthesis
epoxygenase_P450_pathway
jasmonic_acid_biosynthesis
flavonoid_biosynthesis
chromatin_silencing
signal_transduction
protein_transport
vesicle_budding

Figure 3: Correlations of gene expression values in wild-type experiments.

7 Conclusions

We have proposed a new general method for clustering labelled graphs that entails (i) identifying interesting substructures, (ii) clustering the graphs by their substructures and (iii) finding frequently co-occurring substructures pairs.

GraphClust performs efficiently and gives high quality results for both artificial and real data sets across a wide variety of domains. GraphClust is implemented in ANSI C and the software implementation is freely available at www.cs.nyu.edu/shasha/papers/graphclust/. Suggestive biological results have been obtained by using GraphClust substructure creating method on the correlations of gene expression values in both wild-type and mutant experiments.

8 Future Work

We are extending GraphClust to deal with nearest neighbor and range query search. Also, a new version that makes use of the Berkeley Database to reduce the space complexity is in progress. At the algorithmic level, we are working on new techniques for

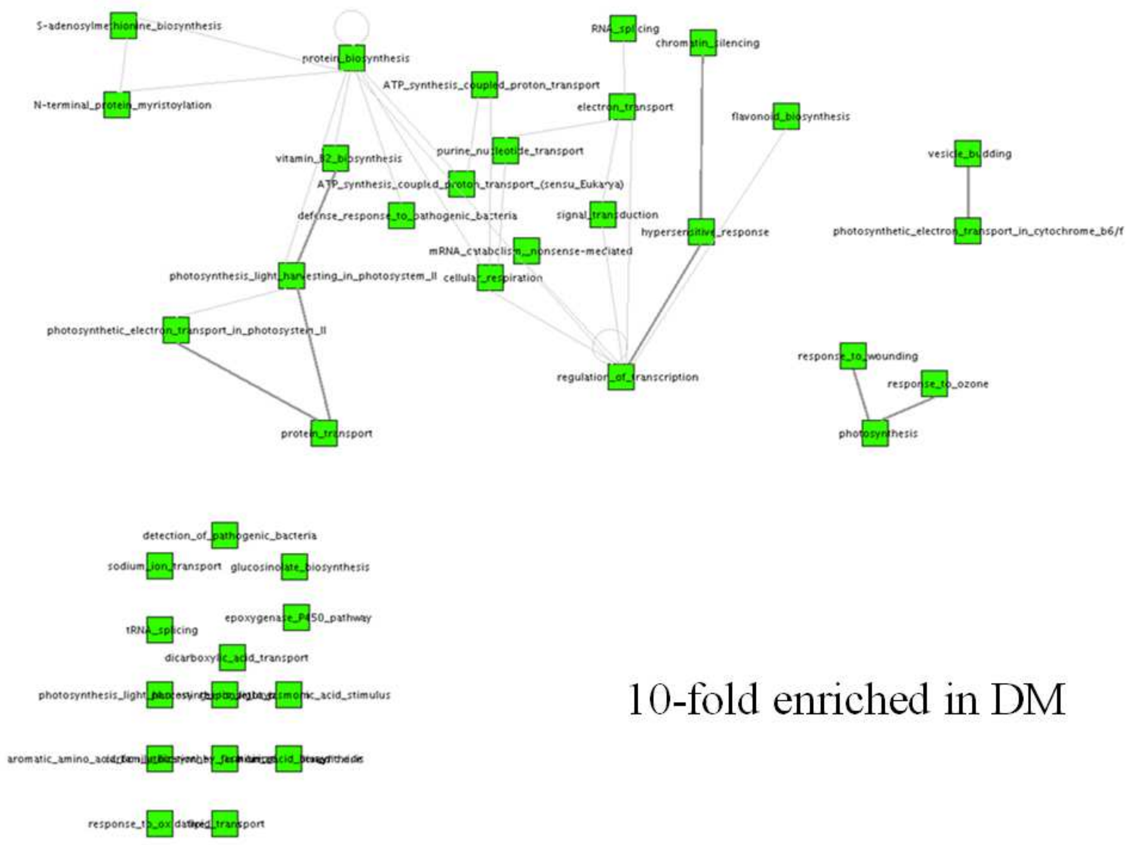
S-adenosylmethionine_biosynthesis
protein_biosynthesis
N-terminal_protein_myristoylation
RNA_splicing
chromatin_silencing
ATP_synthesis_coupled_proton_transport
electron_transport
flavonoid_biosynthesis
vesicle_budding
purine_nucleotide_transport
vitamin_B2_biosynthesis
ATP_synthesis_coupled_proton_transport_(sensu_Eukarya)
defense_response_to_pathogenic_bacteria
signal_transduction
hypersensitive_response
photosynthetic_electron_transport_in_cytochrome_b6/f
mRNA_catabolism,_nonsense-mediated
photosynthesis_light_harvesting_in_photosystem_II
cellular_respiration
photosynthetic_electron_transport_in_photosystem_II
response_to_wounding
regulation_of_transcription
response_to_ozone
protein_transport
photosynthesis
detection_of_pathogenic_bacteria
sodium_ion_transport
glucosinolate_biosynthesis
epoxygenase_P450_pathway
tRNA_splicing
dicarboxylic_acid_transport
response_to_ox
10-fold enriched in DM

Figure 4: Correlations of gene expression values in nitrate reductase (NR)-null double mutant of Arabidopsis experiments.

finding common substructures and classifying the most important ones. A new GraphClust extension will make use of data mining tools like k-min-hashing for identifying important substructures. As far as bioinformatics is concerned, we are looking for some way to incorporate GraphClust in Cytoscape [1] and doing a further analysis to protein-protein interaction networks.

Acknowledgement

We thank Dr. Rodrigo A. Gutiérrez from the Department of Biology of New York University for having provided us the biological datasets, giving extensive comments and important suggestions. This work has been partly supported by the U.S. National Science Foundation under grants NSF IIS-9988345, N2010-0115586, and MCB-0209754. This support is greatly appreciated.

References

[1] Cytoscape. http://www.cytoscape.org.

[2] The subdue knowledge discovery system. http://cygnus.uta.edu/subdue.

[3] M.W. Berry, Z. Drmac, and E.R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Review*, 41(2):335–362, 2003.

[4] B.Luo, R.C.Wilson, and E.R.Hancock. Spectral feature vectors for graph clustering. *Proceedings of joint Syntactical and Structural Pattern Recognition and Statistical Pattern Recognition*, 2396:83–93, 2002.

[5] B.Luo, R.C.Wilson, and E.R.Hancock. Spectral clustering of graphs. *Proceedings of 4th IAPR-TC15 Graph based Representations in Pattern Recognition*, pages 190–201, 2003.

[6] N. Bolshakova and F. Azuaje. Improving expression data mining through cluster validation. *Information Technology Applications in Biomedicine, 2003*, pages 19–22, 2003.

[7] A. Brint and P. Willett. Algorithms for the identification of threedimensional maximal common substructures. *J. Chem. Inf. Comput. Sci.*, 27:152–158, 1987.

[8] R.D. Brown and Y.C. Martin. Use of structure activity data to compare structure-based clustering methods and descriptors for use in compound selection. *J. Chem. Inf. Comput. Sci.*, 36:572–584, 1996.

[9] R.D. Brown and Y.C. Martin. The information content of 2d and 3d structural descriptors relevant to ligand - receptor binding. *J. Chem. Inf. Comput. Sci.*, 37:1–9, 1997.

[10] H. Bunke. Graph-based tools for data mining and machine learning. *Proceedings of Machine Learning and Data Mining in Pattern Recognition*, pages 7–19, 2003.

[11] D. Cantone, A. Ferro, A. Pulvirenti, D. Reforgiato, and D. Shasha. Antipole tree indexing to support range search and k-nearest-neighbor search in metric spaces. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 17(4):535–550, 2004.

[12] J. Clark and S. DeRose. http://www.w3.org/TR/xpath, 1999.

[13] D.R. Cutting, D.R. Karger, J.O. Pedersen, and J.W. Tukey. Scatter / gather: A cluster-based approach to browsing large document collections. *Proc. ACM SIGIR 92*, pages 318–329, 1992.

[14] S. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, and R.A. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.

[15] G.M. Downs and J.M. Barnard. Clustering methods and their uses in computational chemistry. *Reviews in Computational Chemistry, Lipkovitz K., Boyd D.B., Eds; VCH Publishers: New York*, 18:1–40, 2002.

[16] M.F.M. Engels, T. Thielmans, D. Verbinden, J.P. Tollenaere, and R. Verbeeck. Cerberus: A system supporting the sequential screening process. *J. Chem. Inf. Comput. Sci.*, 40:241–245, 2000.

[17] A. Ferro, R. Giugno, A. Pulvirenti, D. Reforgiato Recupero, and D. Shasha. Blastgen, a graphs generator for graph matching benchmarking. *Preprint*, 2005.

[18] E. Forgy. Cluster analysis of multivariate data: Efficiency vs interpretability of classifications. *Biometrics*, 21:767–780, 1965.

[19] L. Galanis, E. Viglas, D.J. DeWitt, J.F. Naughton, and D. Maier. Following the paths of xml data: An algebraic framework for xml query evaluation. *Submitted*, 2001.

[20] R. Giugno and D. Shasha. Graphgrep, a fast and universal method for querying graphs. *Proceeding of the IEEE International Conference in Pattern recognition (ICPR)*, 2002.

[21] T.F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[22] T.F. Herpin, K.G. Van Kirk, J.M. Salvino, and S.T. Yu. Synthesis of a 10000 member 1,5 benzodiazepine-2-one library by the direct sorting method. *J. Chem. Inf. Comput. Sci.*, 2:513–521, 2000.

[23] David Hull. Improving text retrieval for the routing problem using latent semantic indexing. *In Proceedings of the 17th ACM/SIGIR Conference*, pages 282–290, 1994.

[24] P. Husbands, H. Simon, and C. Ding. On the use of singular value decomposition for text retrieval. *Proc. of SIAM Comp. Info. Retrieval Workshop*, pages 145–156, 2001.

[25] C.A. James, D. Weininger, and J. Delany. Daylight theory manual-daylight 4.71. *Daylight Chemical Information Systems, www.daylight.com*, 2000.

[26] R.A. Jarvis and E.A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Trans. Comput.*, C-22:1025–1034, 1973.

[27] M.A. Johnson and G.M. Maggioara. Concepts and applications of similarity. *Wiley-New York*, 2000.

[28] S. Kosinov and T. Caelli. Inexact multisubgraph matching using graph eigenspace and clustering models. *Proceedings of joint Syntactical and Structural Pattern Recognition and Statistical Pattern Recognition*, 2002.

[29] G. Kowalski. *Information retrieval systems: Theory and implementation.* Boston: Kluwer Academic Publishers, 1997.

[30] Steinbach M., Karypis G., and Kumar V. A comparison of document clustering techniques. *Proc. Text Mining Workshop, KDD 2000*, pages 1–11, 2000.

[31] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. Lore: A database management system for semistructured data. *SIGMOD Record*, 26:54–66, 1997.

[32] P.B. Menard, R.A. Lewis, and J.S. Mason. Rational design and compound selection: Cascaded clustering. *J. Chem. Inf. Comput. Sci.*, 38:497–505, 1998.

[33] B.T. Messmer and H. Bunke. Subgraph isomorphism detection in polynomial time on preprocessed model graphs. *Proceedings of ACCV*, pages 373–382, 1996.

[34] T. Ott, A. Kern, A. Schuffenhauer, M. Popov, P. Acklin, E. Jacoby, and R. Stoop. Sequential superparamagnetic clustering for unbiased classification of high-dimensional chemical data. *J. Chem. Inf. Comput. Sci.*, 44:1358–64, 2004.

[35] S. Rajappap. Interactive biasing in graph-based data mining. *Master Thesis in Computer Science and Engineering*, 2003.

[36] J. Shanmugasundaram, H. Gang., K. Tufte, C. Zhang, D. De Witt, and J.F. Naughton. Relational databases for querying xml documents: Limitations and opportunities. *VLDB Journal*, 1999.

[37] N.E. Shemetsukis, J.B. Dunbar, B.W. Dunbar, D.W. Moreland, and C. Humblet. Enhancing the diversity of a corporate database using chemical database clustering and analysis. *J. Comput.-Aided Mol. Des.*, 9:407–416, 1995.

[38] L. Sheng, Z.M. Ozsoyoglu, and G. Ozsoyoglu. A graph query language and its query processing. *ICDE*, pages 572–581, 1999.

[39] D. Suciu. An overview of semistructured data. *SIGACTN: SIGACT News (ACM Special Interest Group on Automata and Computability Theory)*, 29, 1998.

[40] J. Wang, B. Shapiro, and D. Shasha. Pattern discovery in biomolecular data. *New York Oxford, oxford university press edition*, 1999.

[41] R. Wang, St. LaBrie, and Nm. Crawford. Genomic analysis of a nutrient response in arabidopsis reveals diverse expression patterns and novel metabolic and potential regulatory genes that are induced by nitrate. *Plant Cell*, 12:1491–1510, 2000.

[42] R. Wang, M. Okamoto, X. Xing, and Nm. Crawford. Microarray analysis of the nitrate response in arabidopsis roots and shoots reveals over one thousand rapidly responding genes and new linkages to glucose, trehalose-6-p, iron and sulfate metabolism. *Plant Physiol*, 132:556–567, 2003.

[43] R. Wang, R. Tischner, R.A. Gutierrez, M. Hoffman, X. Xing, M. Chen, G. Coruzzi, and N. Crawford. Genomic analysis of the nitrate response using a nitrate reductase-null mutant of arabidopsis. *Plant Phisiol*, 136:2512–2522, 2004.

[44] Y.H. Wang, Gf. Garvin, and Lv. Kochian. Nitrate-induced genes in tomato roots: array analysis reveals novel genes that may play a role in nitrogen nutrition. *Plant Physiol*, 127:345–359, 2001.

[45] J.H. Ward. Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.*, 58:236–244, 1963.

[46] D. Wild and C.J. Blankley. Comparison of 2d fingerprint types and hierarchy level selection methods for structural grouping using ward's clustering. *J. Chem. Inf. Comput. Sci.*, 40:155–162, 2000.