Extracting and Mining Protein-Protein Interaction Network from Biomedical Literature

Xiaohua Hu, Illhoi Yoo, Il-Yeol Song, Min Song, Jianchao Han, Mark Lechner

Abstract—**Protein-protein interaction network study has attracted a lot of attention from bioinformatics community because it is essential to understand the fundamental processes that govern cell biology. However, most of the protein-protein interaction information relevant to cell biology research still exists only in biomedical literature, which is written in a natural language that computers cannot easily manipulate. Retrieving and mining this information is very difficult due to the lack of formal structure in the natural-language narrative in those documents and the huge volume of biomedical literature. In this paper we integrate information extraction and data mining techniques to extract and mine the protein-protein interaction network from biomedical literature such as MedLine. Our system SPIE-DM (Scalable and Portabel Information Extraction and Data Mining) consists of two phases: Phase 1: we develop a Scalable and Portable IE method (SPIE) to extract the protein-protein interaction from the biomedical literature. These extracted protein-protein interactions form a scale-free network graph. In Phase 2, we apply a novel clustering method SFCluster to mine the protein-protein interaction network. The clusters in the network graph represent some potential protein complexes, which are very important for biologist to study the protein functionality. The clustering algorithm considers the characteristics of the scale-free network graphs and is based on the local density of the vertex and its neighborhood functions that can be used to find more meaningful clusters at different density levels. The experiments of SPIE-DM on around 1600 chromatin proteins indicate that our system is very promising for extracting and mining from biomedical literature databases.**

Index Terms—**graph, protein-protein interaction, text mining**

I. Introduction

Many biological results are published only in plain–text documents and these documents or their abstracts are collected in online biomedical literature databases such as MedLine. MedLine contains bibliographic citations and author abstracts from more than 4,000 biomedical journals and is the largest English language biomedical bibliographic database with more than 12 million abstracts stored in plain text files. The sheer size of MedLine can be daunting to many scientists involved in biomedical research. To expedite the progress of functional bioinformatics, it is important to develop scalable learning methods to efficiently process large amounts of biomedical literature and to extract the results into a structured format that is easy for retrieval and analysis by genomic and medical researchers. Automated discovery and extraction of these biological relationships from biomedical literatures have become essential because of enormous amounts of biomedical literature published each year. A promising approach for making such huge amounts of information manageable and easily accessible is to integrate information extraction and data mining methods to automatically process biomedical literature and to extract important biological relationships such as protein–protein interactions and to consolidate them into a structured format such as databases. Some databases that accumulate these biological relationships are DIP for protein–protein interactions [1] and BIND for molecular interactions [2]. Most of the protein-protein interaction relationships stored in these databases are manually constructed. However, it is becoming more and more difficult for curators to keep up with an increasing volume of literature. Thus, automatic methods are needed to speed up the construction of such databases. Integration of data mining and IE provides a promising direction to assist in the curation process to construct such databases. Biomedical literature mining has recently attracted a lot of attention from IE, data mining, natural language processing (NLP) and bioinformatics community [3]. A lot of methods have been proposed and various systems have been developed in extracting biological relationships from biomedical literature such as finding protein or gene names [4], protein–protein interactions [5], *etc*.

Scalability and portability are two major problems which are recognized as impeding widespread use of IE in huge collections of biomedical literature such as MedLine. With the development of genomic research, the scope and goal of bioinformatics research is getting more complicated and the number of published documents is growing at a very fast rate, thus the IE and mining methods must be flexible to work for multiple goals in different sub–disciplines and should be able to scale to millions of documents. Current IE techniques extract biological relations from MedLine by examining every abstract, or use filters to select promising abstracts for extraction. With more than 12 million abstracts in MedLine database, processing time is becoming a bottleneck when exploiting IE technology for leveraging extracted information with relational databases. Examining every abstract is not feasible for huge online biomedical literature databases. Filtering techniques focus on potentially useful abstracts and can dramatically improve the efficiency and scalability of the IE process. However, the current filtering techniques require

Manuscript received June 14, 2004.

Xiaohua Hu, Illhoi Yoo, Il-Yeol Song and Min Song are with College of Information Science and Technology, Drexel University, Philadelphia, PA, 19104 (the first author's e-mail is thu@cis.drexel.edu). Hu's work is supported by rsearch grants from the PA Dept. of Health

Jianchao Han is with Department of Computer Science, California State University

Mark Lechner is with Department of BioScience & BioTech, Drexel University, Philadelphia, PA 19104

human involvement to maintain and to adapt to new topics or sub disciplines.

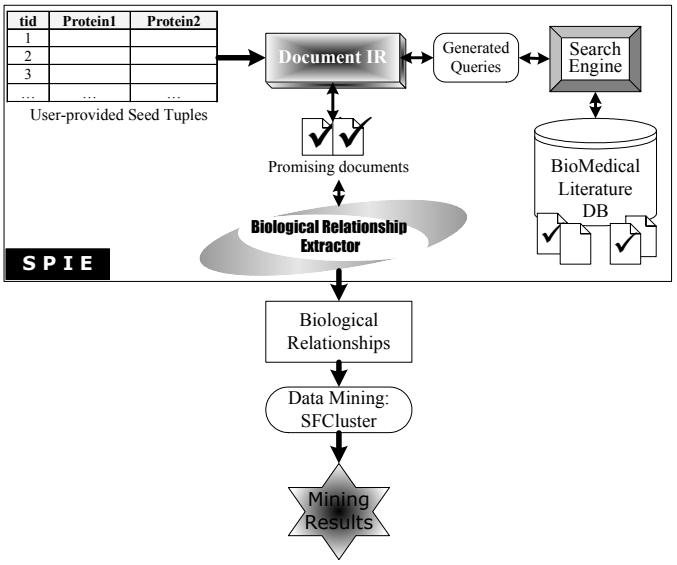

Fig. 1. Architecture of SPIE-DM

This paper discusses a hybrid system SPIE-DM (Scalable and Portable IE-Data Mining), which integrates information extraction and data mining to automatically extract and mine biological relationships from a huge collection of biomedical literature to help biologists in functional bioinformatics research. The system architecture is shown in Figure 1. SPIE-DM consists of two phases: Phase 1: SPIE is used to extract the protein-protein interaction from the biomedical literature. These extracted protein-protein interactions form a scale-free network graph that has many distinct properties such as in-degrees and out-degrees of the vertices following power laws. In Phase 2, we apply a novel clustering method SFCluster to mine the protein-protein interaction network. The clusters in the network graph represent some potential protein complexes, which are very important for biologist to study the protein functionality. The clustering algorithm considers the characteristics of the scale-free network graphs and is based on the local density of the vertex and its neighborhood functions that can be used to find more meaningful clusters with different density level.

By using an automated information extraction approach, biological relationships and knowledge such as protein-protein interaction from scattered sources will be synthesized and consolidated into a single database and are amenable to computational analysis. Compared with previous work, our methods reduce the manual intervention to a great minimum. The most closely related work to ours is Snowball [6]. Snowball can only handle situations where one entity is involved in only one relationship in bioinformatics domain; however, an entity may be involved in many relationships. For example, a protein may interact with many other proteins.

By mining the protein-protein interaction network, indirect or hidden relationships between proteins will be identified and clustered. This will reveal information beyond simple binary interactions about important protein complexes or pathways, which otherwise would prove to be challenging and time-consuming by manual searches of the literature. These biological relationships will help uncover hidden relationships and complexes governing genomic operations.

The rest of the paper is organized as follows: In Section 2, we discuss the technical details of SPIE. We present our novel cluster algorithm SFCluster in Section 3. We conclude with some discussion and our future research plan in Section 4.

II. Principle and Architecture of SPIE for Information Extraction

SPIE uses a pipelined architecture and extracts with as little human intervention as possible. Unlike previous approaches, which use annotated corpus for training, we only use a few seed examples, making it easier to port from one subject domain to another. In biomedical research, especially in rapidly changing fields such as molecular biology and medicine, subjects can be extremely complex: there are many synonym terms, new connections are constantly discovered between previously unrelated subjects, and review documents are outdated very quickly. In these situations, a technique in which queries are automatically updated based on the previous search results is necessary in order to retrieve relevant documents from large text databases for IE. Such an automatic query learning technique allows an IE system to be easily adapted to a new domain or to a new databases with minimal human effort. Based on these considerations, we develop a scalable and portable information extraction method with automatic query learning system. The data flow architecture of SPIE is shown in Fig. 2.

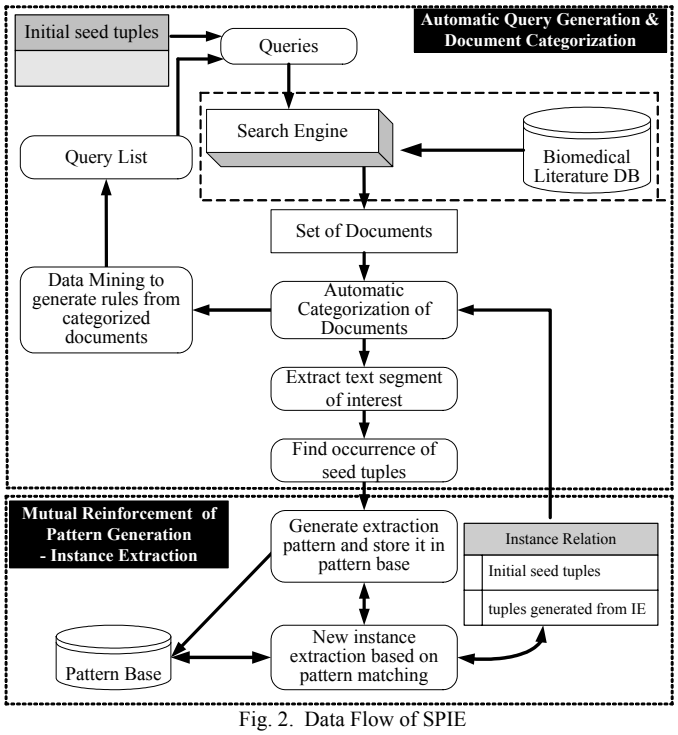

Fig. 2. Data Flow of SPIE

SPIE consists of the following steps:

1. Starting with a set of user-provided seed tuples (the seed tuples can be quite small, normally 5 to 10 is enough), SPIE retrieves a sample of documents from the biomedical

literature library. At the initial stage of the overall document retrieval process, it has no information about the documents that might be useful for the goal of extraction. The only information we require about the target relation is a set of user-provided seed tuples, including the specification of the relation attributes to be used for document retrieval. We construct some simple queries by using the attribute values of the initial seed tuples to extract the document samples of a pre-defined size using from the search engine.

2. The tuple set induces a binary partition (a split) on the documents: those that contain tuples or those that do not contain any tuple from the relation. The documents are thus labeled automatically as either positive or negative examples, respectively. The positive examples represent the documents that contain at least one tuple. The negative examples represent documents that contain no tuples.
3. Next data mining and/or IR techniques are applied to the classified documents obtained from Step 2 to derive queries targeted to match—and retrieve— additional documents similar to the positive examples.
4. Then a mutual reinforcement pattern generation and tuple extraction technique is applied over the documents. It produces a set of extracted patterns using the top-ranked tuples in the relations. These patterns are kept in the pattern base, which will be used to generate new tuples from the documents.
5. The system queries the biomedical literature databases using the automatically learned queries from Step 3 to retrieve a set of new promising documents from the databases and then goes to Step 2. The whole procedure repeats until no new tuples can be added into the relation or the number of text documents to be processed has reached the pre-set limit.

The technical details of some key steps are discussed in the following subsections

A. Learning Queries to Retrieve Potential Promising Documents

Previous approaches for addressing the high computational cost of IE techniques used document filtering techniques which select the documents that deserve further processing by the IE system. This filtering still requires scanning the complete database to consider every document. Alternative approaches use keywords or phases as filter (which could be converted to queries) that were manually crafted and tuned by the IE system developers. In biomedical and bioinformatics domain, there exist research topics that cannot be uniquely characterized by a set of key words because relevant keywords are (i) also heavily used in other contexts and (ii) often omitted in relevant documents because the context is clear to the target audience. To yield a high recall at a reasonable precision, the results of a broad information retrieval search have to be filtered to remove irrelevant documents. We use automated text categorization for this purpose. In the initial round, we select a pre–specified number of documents based on the seed examples. For example, if our system is used for extraction of protein–protein interactions, the seed examples are a set of protein name pairs as shown in Table 1.

TABLE I
INITIAL TRAINING SEED TUPLES

Protein 1	Protein 2	Interaction
HP1	histoneH3	Yes
HP1	HDAC4	Yes
KAP1	SETDB1	Yes
AuroraB	INCENP	Yes

Therefore, we can first select some of those documents in MedLine which contains at least one pair of related protein name in the seed examples. If a document does contain a pair of protein names in a single sentence, we label it as a positive document; otherwise a negative one. These labeled documents are used in the later stage for data mining algorithms to learn the characteristic of the documents, and the learned rules are converted to a query list in order to retrieve potentially promising documents for IE in the next iteration. Starting from the second round, we use the query list derived from the learned rules to select potentially interesting documents and rely on all the available tuples for document classification. The procedure is illustrated in Figure 3.

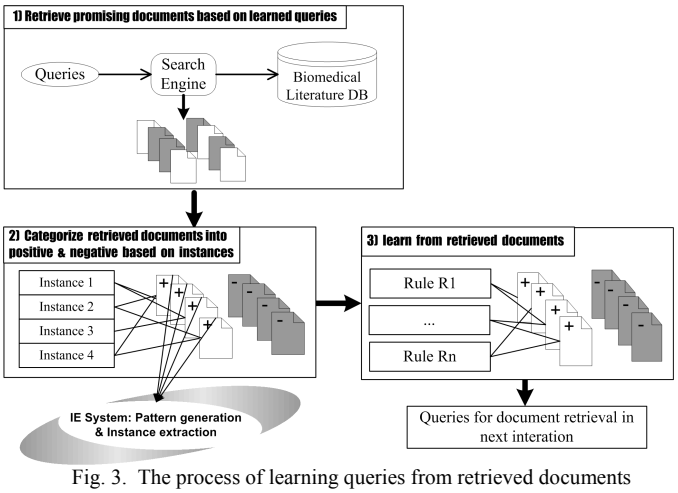

Fig. 3. The process of learning queries from retrieved documents

Given a set of positive and negative documents as the training set, our goal now is to generate queries that would retrieve many documents that our IE system will find useful, and few that IE will not be able to use. The process consists of two stages: (1) convert the positive and negative examples into an appropriate representation for training, and (2) run the data mining algorithms on the training examples to generate a set of rules and then convert the rules into an ordered list of queries expected to retrieve new useful documents. In our current implementation, we integrate fast rule induction algorithms: Ripple [7]. We rank all the rules based on the Laplace measures and the top 10% of the rules are converted into a query list.

For example if a rule set is

```
  Positive  IF WORDS  ~
protein AND binding .
  Positive  IF WORDS ~
cell and function .
```

then they can be converted to a query list:

```
Query 1:  protein AND binding
```

Query 2: cell AND function

Unlike most other IR systems which uses a single term selected with statistical-based term weighting [8], we use data mining algorithms to extract rules from documents and then use the terms from the rules as the basic unit for our next query term.

B. Mutual Reinforcement Principle for Pattern Generations and Tuple Extraction

A crucial step in the extraction process is the generation of new patterns. Patterns are generated by grouping the occurrences of known patterns in documents that occur in similar contexts. Good patterns should be selective but have high coverage so that they do not generate many false positive and can identify many new tuples. Most machine learning methods and algorithms have been developed to automatically generate extraction patterns. These methods and algorithms use special training resources, such as texts annotated with domain-specific tags (e.g., AutoSlog [9], WHISK [10]). A key limitation of using machine learning methods to induce IE methods is the availability of high-quality pre-classified corpora in IE from text database. Creating a pre-classified corpus entails high workload for domain experts, and a corpus for a specific domain cannot usually be directly transferred to other domains, thus making the portability a very challenging issue. The heart of our approach is the mutual reinforcement technique that learns extraction patterns from the tuples and then exploits the learned extraction patterns to identify more tuples that belong to the relation.

The pattern representation used in SPIE is similar to those used in Eliza [11], which can make use of limited syntactic and semantic information. SPIE represents the context around the related entities in the patterns in a flexible way that produces patterns that are selective, yet have high coverage.

Definition 1 A pattern is a 5–tuples *<prefix, entity_tag1, infix, entity_tag2, suffix>*, where *prefix*, *infix*, and *suffix* are vectors associating weights with terms *entity_tag1*and *entity_tag2*. *Prefix* is the part of sentence before *entity_tag1*, *infix* is the part of sentence between *entity_tag1* and *entity_tag2* and *suffix* is the part of sentence after *entity_tag2*.

For example, a protein–protein interaction pattern in our approach is represented as a tuple (or an expression) consisting of two protein names that correspond to some conventional way of describing the interaction. For every such a protein pair tuple *<p1, p2>*, it finds segments of text in a sentences where *p1* and *p2* occur close to each other and analyzes the text that "connects" *p1* and *p2* to generate patterns. For example, our approach inspects the context surrounding chromatin protein *HP1* and *HDAC4* in "*HP1 interacts with HDAC4 in the two–hybrid system*" to construct a pattern *{ "", <Protein>, "interacts with", <Protein>, ""}*. After generating a number of patterns from the initial seed examples, our system scans the available sentences in search of segment of text that match the patterns. As a result of this process, it generates new tuples and uses them as the new "seeds" and starts the process all over again by searching for these new tuples in the documents to identify new promising patterns.

In order to learn these patterns from these sentences, we use a sentence alignment method to group similar patterns together and then learn each group separately for the generalized patterns.

Definition 2. The $Match(T_i, T_j)$ between two patterns T_i and T_j, which are two 5–tuples T_i=<$prefix_i$, tag_{i1}, $infix_i$, tag_{i2}, $suffix_i$> and T_j = <$prefix_j$, tag_{j1}, $infix_j$, tag_{j2}, $suffix_j$>, is defined as $Match(T_i, T_j)=W_{prefix}*Sim(prefix_i, prefix_j)+W_{infix} * Sim(infix_i, infix_j) + W_{suffix} *Sim(suffix_i, suffix_j)$

There are many methods or formulas available to evaluate the similarity of two sentence segments such as $perfix_i$ and $prefix_j$, which are ordered list of words, numbers and punctuation marks etc. In our system, we use the sentence alignment function similar to the sequence alignment in bioinformatics. The advantage of using sentence alignment for similarity measurement is that it is flexible and can be implemented efficiently based on dynamic programming.

After generating patterns, SPIE scans the text collection to discover new tuples. The basic ideas are outlined below. Our system first identifies sentences that include a pair of entities. For a given text segment, with an associated pair of entities E_1 and E_2, it generates the 5–tuples T=<*perfix, E_1_tag1, infix, E_2_tag2, suffix*>. A candidate tuple <E_1, E_2> is generated if there is a pattern T_p such that $Match(T, T_p)$ is greater than the pre-specified threshold. Each candidate tuple will then have a number of patterns that helped generate it, where each is associated with a degree of match. Our approach relies on this information, together with score of the patterns (the score reflects the selectivity of the patterns), to decide what candidate tuples to actually add to the biological relationship table that is being constructed. Below are some sample extraction patterns generated from MedLine for protein-protein interactions.

{"", <Protein>, "interacts with", <Protein> , ""}
{" ", <Protein>, "binds to", <Protein> , ""}
{"*Bind of*", <Protein>, "to" ,<Protein>, ""}
{ "Complex of ", <Protein> ,"and ", <Protein>, ""}

To test the scalability of SPIE in MedLine, we conducted two experiments. One is to simulate the biologist to manually create a set of keyword filters to select the documents which are relevant to protein interactions, and then run the IE procedure on these documents. Nowadays this manual approach is used by most users of Medline. However, information retrieval in such databases becomes very time-consuming because searchers who are likely to identify much relevant information also find many irrelevant documents. For example, a text query for "protein interaction" of the Medline database retrieves 145857 documents (in Dec. 2003). In this study, we use 1600 human chromatin protein names. When we used synonyms derived from LocusLink and nucleotide databases maintained by NCBI, the total number of protein names was around 7000. The result is shown in Table 2. In our second experiment, we started with 10 pairs of protein-protein interaction (PPI) pairs as seed instances. We then used SPIE to automatically construct queries and used the learned queries to retrieve

document from MedLine. In each iteration, we set the maximal document size to 10k for each iteration, starting with 50,000 documents and stop at 500,000 documents when the new tuples added is very small. We repeated the experiments 5 times with different seed-pairs and took the average number of documents. The results are summarized in Table 3.

TABLE II
NUMBER OF MEDLINE ABSTRACTS USED IN KEY WORD BASED SEARCHING

Keywords	# of abstracts	# of PPI	# of distinct PPI
Protein Associate	8025	2526	760
Protein Interact	33835	8457	2158
Protein Bind	69981	12034	2664
Protein Association	82767	9440	2093
Protein Binding	83397	13854	3184
Protein interaction	145857	19344	3795
Protein complex	185157	24938	4300
Protein acetylate	172	434	116
Protein acetylation	5027	5622	827
Protein conjugate	18770	225	92
Protein destabilize	879	100	31
Protein destabilization	2233	231	62
Protein inhibit	124178	7690	1602
Protein modulate	41727	2984	945
Protein modulation	71159	2843	913
Protein phosphorylate	3991	1186	315
Protein phosphorylation	90475	15106	2249
Protein regulate	58586	7991	2121
Protein regulation	289940	32669	5915
Protein stabilization	27349	1630	340
Protein stabilize	5714	775	221
Protein suppress	20069	2005	633
Protein target	74714	10735	2433
Total	1,444,002	183,119	37,769
Total (elimination of redundant ones)	1,006,699	37769	9980

TABLE III
EXPERIMENTAL RESULTS (SPIE)

# of abstracts	# of PPI	# of distinct PPI
50k	2224	1749
100k	4412	3100
150k	8348	4400
200k	10527	5300
250k	12461	6040
300k	15152	6500
350k	16612	7200
400k	18202	8420
450k	19070	8900
all	19461	9483

While keyword based approach examined 1.4 millions abstracts from MedLine to extract 9980 distinct chromatin protein-protein interaction, SPIE examined only 500K abstracts from MedLine to extract 9483 distinct chromatin protein-protein interactions. It is very obvious that SPIE has a significant performance advantage over the key-word based approach.

III. MINING THE SCALE-FREE PROTEIN-PROTEIN INTERACTION NETWORK: SFCLUSTER

The extracted protein-protein interactions from MedLine through SPIE form a scale-free network. In a scale-free network, the nodes with the largest numbers of links play an important role on the dynamics of the system. It helps to understand the global structure of the network as well as its precise distribution of the number of links. Recently empirical studies report that the protein-protein interaction network [12], like many other network graphs generated either from the real world or the man-made world such as the Internet, the WWW, have scale-free properties [2], [13], [14], [15], [16] [17]. The scale-free property reveals that the number of incoming links and the outgoing links at a given vertex have distributions that decay with the power law tails [14]. A scale-free network has many vertices but few vertices of high degree. These networks are so complex that the topology is largely unknown. It is essential to mine the network graph to help understand the domain and the topology of the network structure. For example, a local cluster in a biological interaction network for proteins may represent a biological complex [2], [16] which is very important to help understand the protein functionality.

Many graph-based clustering algorithms have been developed to analyze the network graphs [18], [19]. Most of these cluster algorithms rely on some properties such as connectivity of the graph in the context of a random graph model to find the clusters. When these types of clustering algorithms are applied in the scale-free network model, many meaningful clusters couldn't be identified since the connectivity of a random graph has totally different behaviors from a scale-free network. Scale-free networks are known to have large clustering coefficients or clustered regions of the graph and there is a lot of useful information hidden in these graph networks. Based on this consideration, we propose a novel algorithm to find local clusters from a large scale-free network graph. Our algorithm considers the characteristics of the scale-free network graph and finds the local clusters based on the local density of the vertex and its neighborhood vertices.

A. Random Network Model Vs. Scale-Free Network Model

Random network models assume that the probability that two vertices are connected is random and uniform [14], [20]. For a random graph, edges are chosen independently, and thus the distribution of degree decays exponentially. Therefore, for the power law degree distribution, the choice of edge must be correlated. Moreover, random network models also assume that the graph is static, meaning that the graph is never changed over time.

In contrast, scale-free networks have two important properties: growth and preferential attachment, meaning that new vertices and edges are added over time and a new vertex is most likely to be attached to existing vertices with large connectivity. The probability that a new vertex connects to the existing vertices is not uniform, but there is a higher probability to be linked to a vertex that already has a large number of connections. These networks have the power law connectivity distribution $P(k) \propto k^{-r}$. That is, the probability that a vertex is connected to k other vertices is proportional to k^{-r}, where r is a constant and varies for diverse networks. The power law concisely describes skewed distributions of graph properties such as the vertex outdegree. Besides, these power-laws are used to estimate important parameters such as the average neighborhood size, and facilitate the design and the performance analysis of protocols.

B. Cluster Algorithm Based On Local Density And Vertex Neighborhood

Clustering the graph corresponds to finding the cliques of the graph. In graph theory, a clique is a maximal connected subgraph, that is, there is an edge between any pair of vertices in the subgraph. There are two main barriers to finding cliques in the graph to determine clusters. The first barrier is the computation issue, since the problem of finding the maximum clique in a graph is NP-hard. Even finding a good approximation to the maximum clique is also hard. The second barrier to using cliques to determine communities is that the connectivity required in a clique is too strong. It would be too much to expect all members of a community to be linked each other. There are many heuristic algorithms to find clusters from a random graph. In these algorithms, a global path-finding strategy has been used, which cannot be applied in real system where only local information is available. Recent studies of the many network graphs suggest that the structure of the scale-free network graph is actually more complex than the random graph model, and thus cannot be modeled in the random graph model.

Our approach is based on local density of the p-quasi graph [21] and vertex neighborhood information of the modified Gabriel influence region [22]. Our approach follows the similar algorithmic philosophy presented in [2]. The method is based on vertex weighting by local neighborhood density and outward traversal from a locally dense seed to isolate the dense regions according to given parameters. Dense regions of the networks can be found, based solely on connectivity data, many of which correspond to some domain information and knowledge.

Before we discuss the algorithm in details, we cite some important definitions used in the algorithm to make the paper self content.

P-quasi complete graph [21]: A p-quasi complete graph *QC = (V,E)* is a graph such that deg(*v*) >= [p*(|V|- 1)] , for all *v* ∈ *V, p is the connectivity (0<=p<=1).*

Curvature (or core clustering coefficient) on a graph [23]: Each vertex *v* has a curvature which is a function of the number *n* of neighbors (vertices to which it is linked) and the number *t* of triangles (pairs of adjacent neighbors), given by the formula:

$$Curv(v) = t / ((n(n-1)/2)).$$

Because *n(n-1) /2* is the maximum number of triangles that can be drawn on *v* neighbors, *Curv(v)* lies between *0* and *1* if *n > 1* and is undefined otherwise. The *Curv(v)* amplifies the weighting of heavily interconnected graph regions while it removes the less connected vertices that are usually part of an interaction network. A given highly connected vertex, *v,* in a dense region of a graph may be connected to many vertices of low degree. These low degree vertices do not interconnect with the neighborhood of *v* and thus removing them would not reduce the curvature of *v*.

Modified Gabriel influence region [22]: For a non-directed graph *G(V,E),* where V is a set of vertices corresponding to a data set and E is the set of edges, the modified Gabriel influence region of vertices *p* and *q* is defined as

$$\Gamma_{p,q} = B((p+q)/2, d(p,q)/2) \cup B(p, \alpha d(p,q)) \cup B(q, \alpha d(p,q)),$$

where $B(x,r) = \{y: d(x,y) <= r\}$, $d(x,y) = |x-y|$ is the distance between the vertices *x* and *y*, and α is a constant. $(p,q) \in E$ if $\Gamma_{p,q} \cap V = \phi$. α= 0.30 and 0.50 are chosen in our experiments for constructing edges and discovering clusters, respectively.

The modified Gabriel influence region has the local neighborhood information for each vertex. Given a Curvature threshold, the graph can be split into clusters that appear to be meaningful for the underlying domain. To find locally dense regions of a graph, we use a vertex-weighting scheme based on the local clustering coefficient *Curv(v)* which measures "cliquishness" of the neighborhood of a vertex *v* and the Gabriel influence region. There is no standard graph theoretic definition of density, but definitions are normally based on the connectivity level of a graph. Our approach is based on the local density and modified Gabriel influence region (vertex neighborhood information) of the graph. The method starts with a seed with largest weight and outward traversal from a locally dense seed to isolate the dense regions according to the given parameters. Dense regions of a network graph can be found, based solely on connectivity data, many of which correspond to some domain information and knowledge.

We define the *weight* of a vertex *v* as

$$W(v) = Curv(v) * |GR(v)|,$$

where $GR(v) = \bigcup_{(v,w) \in E} \Gamma_{v,w.}$

The *weight* of a vertex *v* is the product of the vertex curvature, *Curv(v)* and *|GR(v)|* , in the immediate neighborhood of the vertex *v*. This weighting scheme further boosts the weight of densely connected vertices. This weighting function is based on the local network density.

Our algorithm SFCluster for discovering clusters is described as follows. We define clusters as connected regions of the graph with the high curvature, which is the local density of triangular relations. The clusters are the densest components of the corresponding graph.

Algorithm: SFCluster (Scale-Free Cluster)

Input: (1) D: a data set, (2) δ: the weight_threshold, (3) p: an integer, (4) Wpct: weight percentage

Output: a list of clusters

1. Construct a Graph *G(V,E)*, where *V* corresponds to the data set *D*, and *E* is calculated based on the modified Gabriel influence region

2. For all *v* in *V* **Do**

/* Calculate the weight of all vertices*/

Find the highest *p-quasi* graph G_p containing *v*

Neb(v) = vertices linked with *v in* G_p

Calculate *Curv(v) in* G_p

$GR(v) = \phi$

For all *w* in *Neb(v)* **Do**

Gabriel Region $GR(v) = GR(v) \cup \Gamma_{v,w}$

EndFor

*Weight (v) = Curv(v) * |GR(v)|*

EndFor

3. AllNode = Sort the vertex in *V* in the decreasing order of *Weight(v), ClusterList* = ϕ

4. While Allnode <> ϕ **Do**

Pick up the first vertex *v* from AllNode

If *Weight(v)* < δ **then** *break*
Cluster(v)= {v} // start a new cluster
Boundary(Cluster(v))=Neb(v)
While Boundary(Cluster(v)) <> ϕ **Do**
W = remove a vertex from Boundary(Cluster(v))
if *Weight(w) > = Weight(v) * Wpct*
then *Cluster (v) = Cluster(v)* $\cup${*w*}
Boundary(Cluster(v))
=Boundary(Cluster(v)) $\cup$ *Neb(w)*
remove w from AllNode
EndWhile
ClusterList = ClusterList $\cup${*Cluster(v)*}
EndWhile
5. Return *ClusterList*

The first step of the algorithm SFCluster is to construct a Gabriel graph based on the modified Gabriel influence region with α= 0.30.

The second step of the algorithm SFCluster, vertex weighting, weights all vertices based on their local network density using the highest p-quasi graph of the vertex and the modified Gabriel influence region with α= 0.50. The highest p-quasi graph is the central of the most densely connected subgraph. We define here the term core-clustering coefficient of a vertex, *v*, to be the density of the highest p-quasi graph immediate neighborhood of v (vertices connected directly to *v*) including *v*. The core-clustering coefficient is used here instead of the clustering coefficient because it amplifies the weighting of heavily interconnected graph regions while it removes the less connected vertices that are usually part of an interaction network. A given highly connected vertex, *v*, in a dense region of a graph may be connected to many vertices of degree one (single linked vertex) in the modified Gabriel influence region. These low degree vertices do not interconnect within the neighborhood of *v* and thus would reduce the clustering coefficient, but not the core-clustering coefficient. The final weight given to a vertex is the product of the vertex core-clustering coefficient and the number of vertices of the modified Gabriel influence region. This weighting scheme further boosts the weight of densely connected vertices.

The third step of the algorithm SFCluster sorts all vertices in the graph in the decreasing order of the vertex weight and initializes the output to empty.

Then, the fourth step is to discover a list of clusters. It takes as input the vertex-weighted graph, picks up the vertex with the maximum weight as a seed of a cluster, and recursively travels outward from the seed. At the beginning, the cluster includes *v* only, and all neighbors of *v* are on the boundary. Each vertex on the boundary is checked to see if its weight is above the given threshold, which is a given percentage away from the weight of the seed vertex. If yes, this vertex is included in the cluster and the cluster boundary is expended to cover its all neighbors. A vertex is taken with the probability proportional to its connectivity. This process stops once no more vertices can be added to the cluster based on the given threshold, and a cluster is found. Then a new cluster is discovered for the next highest weight vertex in the graph. In this way, the densest regions of the network are identified. The vertex weight threshold parameter defines the density of the resulting complex. A threshold that is closer to the weight of the seed vertex identifies a smaller, denser network region around the seed vertex.

The time complexity of the entire algorithm is polynomial $O(nml^3)$, where *n* is the number of vertices, *m* is the number of edges, and *l* is the vertex size of the average modified Gabriel influence region of the graph.

We applied SFCluster on the protein-protein interaction network constructed by SPIE from MedLine based on the 1600 chromatin proteins and it identified many local clusters which correspond to protein complex. Most of the protein complexes are evaluated to have a high agreement rate with the domain expert and 2 of these clusters are shown in Figure 4. These studies will provide an ideal test bed for utilizing and assessing the SFCluster algorithm for clustering in scale-free biological networks. It will help to identify previously unknown links between clusters, or links to pathways and networks not considered "chromatin", these clusters could represent communication between chromatin and other parts of the cells. The figure was created using Pajek [http://vlado.fmf.uni-lj.si/pub/networks/pajek/default.htm].

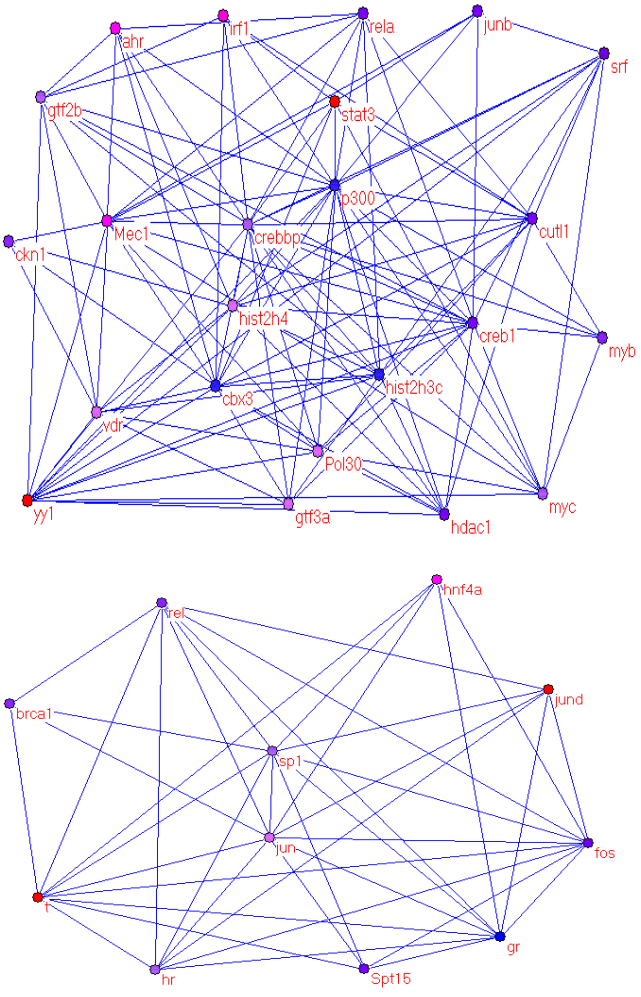

Fig. 4. Three Protein Clusters identified from the Chromatin Protein-Protein Interaction Network

IV. Conclusion

In this paper, we presented SPIE-DM system that integrates information extraction and data mining techniques to extract and mine protein-protein networks from biomedical literature. .First we presented a novel scalable and portable information extraction (SPIE) method to extract biological relationships from biomedical literature. Our method addresses portability and performance issues simultaneously. The SPIE is efficient to work in large online biomedical literature database because it takes an input a few seed examples and automatically iterates to retrieve more relevant documents. Therefore, SPIE is flexible to be applied in very complicated domains and works with little human intervention. SPIE can be used to extract many binary relationships such as protein–protein interactions, cell signaling or protein–DNA interactions from large collection of text files once the name dictionary of the studied object is provided. Our cluster algorithm SFCluster mines dense regions of the large scale-free network graph. Our algorithm considers the characteristics of the scale-free network model and is based on the local connectivity and the modified Gabriel influence regions. Based on our scale-free network analysis, it would seem that real biological networks are organized differently than random graph models in that they have higher clustering coefficients around specific region (complexes) and the vertices in these regions are related to each other by biological function.

Our future research work will be focusing on evaluation of the precision and recall of SPIE-DM. For small biomedical documents sets, it is possible to manually inspect them and calculate the precision and recall. Unfortunately, this evaluation approach does not scale well and becomes infeasible for large collection of literature such as MedLine. Developing accurate evaluation metrics for this task is one of our future research plans.

References

[1] I. Xenarios, E. Fernandez, L. Salwinski, X.J. Duan, M.J. Thompson, E.M. Marcotte and D. Eisenberg. (2001), DIP: The Database of Interacting Proteins: 2001 update, Nucleic Acids Reg., vol. 29(1), pp. 239–241.

[2] G. Bader and C. Hogue. (2003), An automated method for finding molecular complexes in large protein interaction networks. BMC Informatics, 4(1):2.

[3] L. Hirschman J.C. Park, J. Tsujii, L. Wong and C.H. Wu (2002), Accomlishments and challenges in literature data mining for biology, *Bioinformatics* **18** (12) 1553-1561.

[4] K. Fukuda, T. Tsunoda, A. Tamura and T. Takagi. (1998), Toward Information Extraction: identifying protein names from biological papers, Proceddings of the Pacific Symposium on Biocomputing '98, pp. 707–718.

[5] T. Ono, H. Hishigaki, A. Tanigami and T. Takagi. (2001), Automated extraction of information on protein–protein interactions from the biological literature, Bioinformatics, vol. 17(2), pp. 155–161.

[6] E. Agichtein and L. Gravano. (2000), Snowball: Extracting Relations from Large Plain–Text Collections, Proceedings of the 5th ACM International Conference on Digital Libraries, pp. 85–94.

[7] W. Cohen. (1995), Fast effective rule induction, Proceeding of 12th International Conference on Machine Learning, pp. 115-123.

[8] S. Robertson and K. Sparck Jones. (1976), Relevance Weighting of Search Terms, Journal of the American Society for Information Science, vol. 27(3), pp. 129-146.

[9] E. Riloff. (1996), Automatically Generating Extraction Patterns from Untagged Text, Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI–96), pp. 1044–1049.

[10] S. Soderland. (1999), Learning Information Extraction Rules for Semi-structured and Free Text, Machine learning, vol. 34(1-3), pp. 233-272.

[11] J. Weizenbaum. (1966), ELIZA – A Computer program for the study of natural language communications between men and machine, Communications of the ACM, vol. 9(1), pp. 36–45.

[12] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori and Y. Sakaki. (2001), A Comprehensive two-hybrid analysis to explore the yeast protein interactome, Proc Natl Acad Sci, USA, vol. 98, pp. 4569-4574.

[13] R. Albert, H. Jeong and A.L. Barabasi. (2000), Error and Attach Tolerance of Complex Networks, Nature vol. 406, pp. 378-382.

[14] A.L. Barabási and R. Albert. (1999), Emergence of scaling in random network, Science 286, pp. 509-512.

[15] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins and J. Wiener. (2000), Graph structure in the web. Proceedings of the 9th International World Wide Web Conference on Computer networks, pp. 309-320.

[16] X. Hu and H. Han. (2003), Discovering of local clusters from large scale-free network graph, in ACM SIGKDD 2003 2nd Workshop on Fractals, Power Laws and Other Next Generation Data Mining Tools, August, 24-27.

[17] X. Hu X., J. Han and N. Cercone, *Discovering of Cyber Communities from the WWW*, to appear in Proc. of *COMPSAC 2003* , Dallas, TX, Nov. 3-6, 2003.

[18] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins and E. Upfal. (2000), The Web as a graph. Proceddings of the 9th ACM SIGMOD-SIFACT-SIGART Symposium on Principles of Databases Systems, pp. 1-10.

[19] C.R. Palmer, P.B. Gibbons and C. Faloutsos. (2002), ANF: A fast and scalable tool for data mining in massive graphs, Proceedings of the 8th ACM SIGKDD International Conference on Knowledge and Data Mining, pp. 81-90.

[20] P. Erdos and A. Rényi. (1960), On the evolution of random graphs, Publ. Math. Inst. Hungar. Acad. Sci., vol. 5, pp. 17-62.

[21] H. Matsuda, T. Ishihara and A. Hashimoto. (1999), Classifying molecular sequences using a linkage graph

with their pairwise similarities, Theoretical Computer Science, vol. 210(2), pp. 305-325.

[22] R. Urquhart. (1982), Graph theoretical clustering based on limited neighborhood sets. Pattern Recognition, vol. 15(3), pp. 173-187.

[23] J. Rougemont and P. Hingamp. (2003), DNA microarray data and contextual analysis of correlation graphs, BMC Informatics, vol. 4(1):15.