Integrated Querying of Disparate Association and Interaction Data in Biomedical Applications

Shi Qiao
Case Western Reserve University
Cleveland, OH, 44106
sxq18@case.edu

Mehmet Koyutürk
Case Western Reserve University
Cleveland, OH, 44106
mxk331@case.edu

Z. Meral Özsoyoğlu
Case Western Reserve University
Cleveland, OH, 44106
mxo2@case.edu

ABSTRACT
In biomedical applications, network models are commonly used to represent interactions and higher-level associations among biological entities. Integrated analyses of these interaction and association data has proven useful in extracting knowledge, and generating novel hypotheses for biomedical research. For example, integrated mining of clinical similarity among diseases, known disease-gene associations, and molecular interactions among proteins provide insight on prioritizing candidate disease genes. However, since most datasets provide their own schema and query interface, opportunities for exploratory and integrative querying of disparate data are currently limited. In this study, we capitalize on RDF-based representations of biomedical interaction and association data to develop a querying framework that enables efficient processing and flexible specification of graph template matching queries. The proposed framework enables integrative querying of biomedical databases to discover complex patterns of associations among a diverse range of biological entities, including biomolecules, biological processes, organisms, and phenotypes. Our experimental results on the UniProt dataset show the proposed framework can be used to efficiently process complex queries, and identify biologically relevant patterns of associations that cannot be readily obtained by querying each dataset independently.

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: General

General Terms
Algorithms

Keywords
UniProt, RDF, Graph Template Matching

1. INTRODUCTION
In post-genomic biology, networks are commonly used to model molecular interactions, as well as high-level associations among various biological entities. These entities include biomolecules, ligands, cellular functions, functional modules, biological processes, tissues, organisms, and phenotypes. Networks are useful to represent a broad range of interactions among biomolecules, including protein-protein interactions (PPIs) [1], gene co-expression [2], transcriptional regulation [3], metabolic pathways [4], genetic interactions [5], and signaling pathways [6]. Higher level associations represented by networks include gene-disease associations [7], clinical similarity or co-morbidity of diseases [7], disease-drug associations [8], molecular response to drugs [9], functional annotation of genes and proteins, and evolutionary relationships among molecules and organisms [10]. The current state-of-the-art in the querying and analysis of disparate interaction and association data is limited to querying each type of data in isolation, or downloading different datasets in bulk and joining them in house for specific analysis and mining tasks. In other words, it is not straightforward for a researcher to identify or infer indirect associations among biological entities by incorporating data in multiple forms. Recently, heterogeneous network models that incorporate multiple types of interactions and associations have been shown to be effective in the identification of unknown relationships among biomolecules, biological processes, diseases and drugs. The applications of such integrative models include disease gene prioritization [11], drug repositioning [12], and functional annotation of proteins [13].

In this study, we build on the demonstrated promise of heterogeneous network models and develop an RDF (Resource Description Framework) -based querying framework to facilitate exploratory querying of integrated biological networks. Here, the term "integrated biological network" refers to the collection of all known functional, physical and statistical interactions, as well as associations among biological entities. RDF is the first W3C standard for enriching information resources on the web with detailed descriptions (i.e. Meta data). It is the commonly used data model for the linked data, and knowledge bases that are shared and exchanged on the web. An RDF dataset consists of a set of triples, in the form (s,p,o), stating that a subject s has the property p whose value is the object o. RDF data can also be visualized as a graph where subjects and objects are nodes and properties (predicates) are edges. Unique identifiers (URI's) can be used for subjects, properties or objects to uniquely refer to entities, relationships or concepts. Literals can also be used for objects [20]. RDF can easily represent a wide range of data and information from structured, semi-structured, or unstructured sources. Thus, it enables seamless interoperability and integration of the data on the web. Since each RDF triple (edge in graph representation) corresponds to a binary predicate, it lends itself for reasoning and inference based applications as well. Traditional approaches for biomedical applications require querying different biology datasets (e.g., UniProt, MeSH, OMIM, Reactome) using a relational database to build a knowledge base by integrating various query results. In comparison, using an integrated RDF dataset offers several advantages. Namely, creating a knowledgebase using a relational database requires parsers for each dataset, designing a schema, tuning the system. Typically, the file formats, and schema changes occur frequently, which requires costly updates for the parsers, the schema, and the queries to keep the system up-to-date and functional. Using RDF, data is integrated seamlessly, queries do

BCB '15, September 09-12, 2015, Atlanta, GA, USA
 ISBN 978-1-4503-3853-0/15/09 $15.00
DOI: http://dx.doi.org/10.1145/2808719.2808734

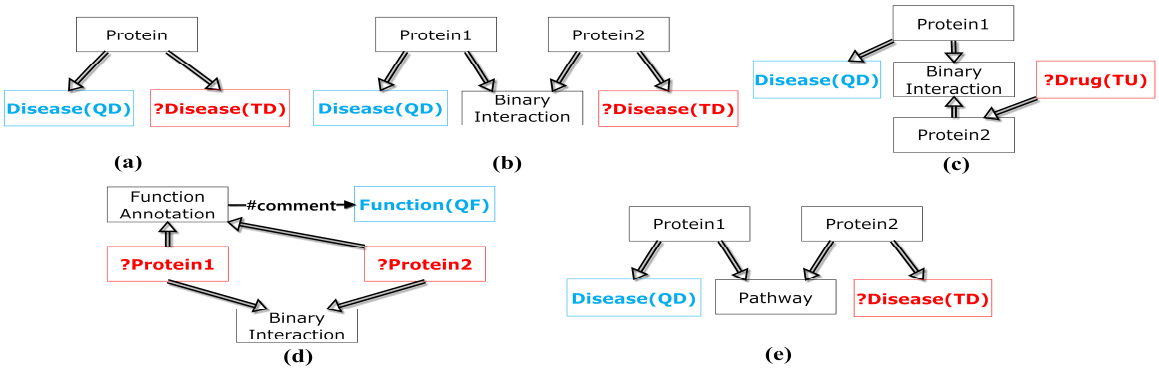


Figure 1. Query Examples and Motivation

(blue nodes represent user specified entities, e.g. "QD" and "QF";
red nodes with "?" represent query output entities, e.g. "?TD" and "?TU"; double line edges represent path connections)

not depend on the schema as in relational databases, and the integrated RDF dataset can be queried directly for interactions and associations between disparate biological entities for information coming from different sources. Here, we propose a graph template-based query framework for querying biological associations and interactions from an RDF knowledge base. In this framework, queries are expressed as a graph template where the nodes are biological entities involved in the query, and edges represent interactions and/or associations between these entities. Our main contribution in this paper is the development of a tool that provides a "shortcut" to data integration in the bioinformatics pipeline. By providing a powerful mechanism for seamlessly querying an RDF dataset integrating various types of biological interactions and associations, this framework enables simple querying and efficient processing for highly sophisticated queries that can provide novel biological insights.

The rest of the paper is organized as follows. In Section 2, we provide examples of using graph template queries for extracting associations and interactions among biological entities that are studied by biomedical scientists. Section 3 provides an overview of RDF data model and Querying RDF using graph templates. In Section 4, we describe the UniProt knowledge base and its components that are utilized in this paper. In Section 5, we present the experimental analysis, queries, and query results. We discuss the related work in Section 6, and conclude in Section 7.

2. QUERY EXAMPLES and MOTIVATION

For a broad range of biomedical applications, researchers query large public databases to interpret their findings, to verify their predictions, or to identify relationships that will help develop novel hypotheses. Common queries that are utilized by biomedical scientists include the exploration of various types of information on a single protein, enrichment analyses for sets of genes or proteins, and identification of genes/protein associated with a given phenotype, function, process, tissue, or drug. However, the current state-of-the-art in integrated mining of biological data shows that disparate databases, when considered together, contain information that cannot be directly extracted by such queries [11-13]. Inspired by the success of integrative data mining efforts, we propose that queries that integrate multiple types of association and integration data will enable scientists to more effectively explore indirect relationships among biological entities. Such queries include:

• **Overlap in the molecular bases of diseases**: The overlap in the identity of genes that are associated with different diseases may be useful in discovering unknown relationships among different diseases [7]. Motivated by this observation, biomedical scientists may be interested in searching for all diseases that share at least a number of gene associations with a disease of interest. Such a question can be addressed by a graph template matching query shown in Figure 1(a). In this query, the user specifies a disease (QD) and queries for all diseases (TD) such that a protein associated with QD is also associated with TD.

• **Shared molecular interactions between diseases**: Diseases with similar molecular etiology may not necessarily overlap in terms of the identity of gene associations, but the relationship may be revealed through molecular interactions among these genes [14]. This provides systems level insights into the shared molecular mechanisms among different diseases [15]. Such relationships can be discovered using a query like the one shown in Figure 1(b). In this query, the user specifies a disease (QD) and queries for all diseases (TD) such that a protein associated with QD interacts with a protein associated with TD.

• **Shared molecular interactions between diseases and drugs**: Drug repositioning has recently become a prominent application in computational biology [16]. This is due to the need to repurpose established drugs to reduce cost, as well as the opportunity provided by omic data to discover unknown relationships between biomolecules targeted by drugs and biological processes involved in pathogenesis. Ability to query for drugs that share a number of molecular interactions with a given disease (i.e., molecular interactions between the targets of the drug and the genes associated with the disease) can provide an excellent starting point for identifying candidate drugs for repositioning [8]. A graph template matching query that can be used for this purpose is shown in Figure 1(c). In this query, the user specifies a disease (QD) and queries for all drugs (TU) such that a protein associated with QD interacts with a protein associated with TU.

• **Network schemas**: Integrated mining of molecular interaction networks and functional annotations led to the identification of network schemas, i.e., small subgraphs of functional terms that recur frequently in molecular interaction networks [18]. These network schemas provide insights into conserved functional modules and the design principles of cellular networks [19]. A graph template matching query such as the one in Figure 1(d) can be used to search for network schemas involving specific biological processes or molecular functions. In this query, the user specifies a specific molecular function (QF) and queries for all proteins associated with QF that interact with each other.

• **Shared pathways between diseases:** Organization of systems biology knowledge in the form of pathways provides well-established, reliable, and tractable access to state-of-the-art

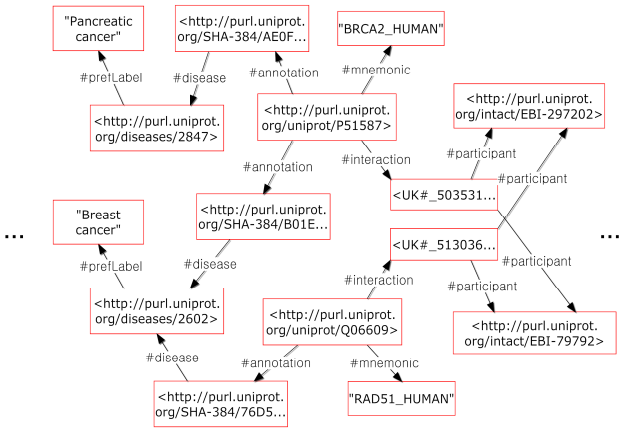


Figure 2. UniProtKB RDF Graph Example

knowledge on biological processes. For this reason, incorporation of pathway data in analyzing the relationship among phenotypes provides information that is complementary to high-throughput interaction data that is organized into networks [17]. For example, identification of shared pathways among different diseases can be useful in understanding similarities in disease development and progression [19]. A sample graph template matching query that can be used for this purpose is shown in Figure 1(e). In this query, the user specifies a disease (QD) and queries for all diseases (TD) such that a pathway that contains a protein associated with QD also contains a protein associated with TD.

As demonstrated by these examples, the ability to seamlessly query interactions and associations among biological entities enables biomedical scientists to get quick answers to a broad range of sophisticated questions on the relationships between these entities. While the examples above are mostly limited to mining indirect associations, it is also possible to use the proposed framework to identify relationships with mechanistic biological interpretations. For example, experimental data derived from gene knock-out or RNA interference experiments are commonly used to identify associations among gene and proteins in terms of their effect on phenotype [21, 22]. For signaling proteins, strong positive or negative correlation between two proteins' influence on the phenotype may be indicative of common downstream effects [23]. Scientists can quickly discover potential candidates for these common effects by querying for paths that go through these proteins and converge into the same node in the integrated network of protein-protein and transcriptional regulatory interactions. Today, the most common way of interpreting observed experimental associations among a group of proteins is to identify subnetworks that connect the proteins of interest, e.g., using Steiner tree based algorithms [24]. Biomedical researchers often use commercial software (e.g., Ingenuity Pathway Analysis, Pathway Studio) that do not provide algorithmic transparency or clearly defined criteria for the identified subnetworks. To this end, semantically meaningful queries that integrate multiple data types can generate significant mechanistic insights into the relationships among proteins of interest and provide the researchers with new ways of thinking about their research questions.

While all of the sample queries listed above can be currently processed by downloading bulk datasets from multiple databases and subsequent processing and joining of these datasets, this is often time-consuming and challenging for many biomedical scientists. Therefore, the main contribution of this study is the development of a querying framework that provides a "shortcut" to data integration in the bioinformatics pipeline. This is useful for two types of applications: 1) "targeted" queries, that is when the researcher is interested in identifying new associations for one or more specific entities (e.g.., a group of genes, a particular disease, a particular drug). 2) "high-throughput" queries for mining tasks, that is when the researcher is interested in identifying all associations that exhibit a specific pattern (e.g., all disease-drug pairs that share a reasonably large number of interactions).

3. RDF QUERY FRAMEWORK

We present the RDF query framework here; see [42] for more details. We first give the basic definitions:

An ***RDF Graph*** is a directed graph $G = \{V, E, l, f\}$ where V is a set of vertices representing either subjects, objects or both. $E \subseteq V \times V$ is a set of directed edges representing predicates pointing from subjects to objects. l is a label set for subjects, objects and predicates. $f: V/E \rightarrow l$ denotes the mapping function between vertices/edges to labels.

Connection edge ($\overset{E}{\Leftrightarrow}$) represents a path $\omega_{i,j}$ between two nodes n_i and n_j. Expression E describes the distance constraints of $\omega_{i,j}$ (Distance is the length of the shortest path). In Figure 3(c), the edge between node "<http://purl.uniprot.org/uniprot*" and node "breast cancer" is a connection edge representing a path of length 4 or less.

A ***GBE (Graph by Example) Query Template*** is a directed graph $G_q = \{V, E\}$ for an RDF graph G, where vertices (objects or subjects) are labeled by partial keywords (that are substrings of labels in the label set l of RDF graph G), and edges represent predicates or connection edges.

Given RDF graph $G = \{V, E, l, f\}$ and query template $G_q = \{V, E\}$, ***Template Matching*** finds all subgraphs of G that satisfy both structural and label constraints in G_q based on graph isomorphism.

A small subset of the UniProtKB RDF dataset represented as a graph is shown in Figure 2, which contains two proteins (BRCA2 and RAD51) and two cancers (Pancreatic cancer and Breast cancer). In general, proteins are associated with diseases through disease annotation nodes. Protein-Protein interactions are from IntAct dataset. Note that two IntAct EBI resources need to be identified for a binary protein–protein interaction in the UniprotKB dataset. This is due to the quality requirements of UniProtKB that only binary interactions which are experimentally supported by multiple observations are imported from the IntAct dataset. Disease information comes from the OMIM dataset. By using this example graph, we demonstrate RDF representations of the association and interaction among different biological entitles from various datasets. We define graph template matching with GBE (graph by example) query templates which support paths, distance constraints, and partial matching of keywords. An example GBE query template is shown in Figure 3 (c) which finds any cancer associated with a protein that is also associated with "Breast cancer" (this query is also shown in Figure 1 (a) and corresponds to query Q3 in Section 5). Evaluating this query over the example RDF dataset shown in Figure 2 will return "Pancreatic cancer" as the answer since the protein BRCA2 is the protein associated with both diseases.

3.1 INDEXES

We utilize two indexes to evaluate graph template queries with partial keywords and connection edges efficiently. The first index is IDMap, which maps RDF labels into integer IDs in lexicographic order. For partial keywords specified as prefixes of RDF labels, the look-up time is $O(logN)$, where N is the total number of RDF

Algorithm: Neighborhood Check (n_i, q_j)	
Input: $n_i \in V(G)$, $q_j \in V(G_q)$, ID Intervals $\mathbb{Z}^*$, {Distance, Count} ψ_j^*, NI Index NI_i for node n_i	
Output: If n_i pass neighborhood check of q_j, return **true**; Otherwise, return **false**	
1	FOR all k *where* $\|k\| \leq d_{max}$
2	FOR any partial keyword $\wp_k$
3	FOREACH value pair $\{d, c\}$ in ψ_j^k
4	Exact all entries from NI_i where ID interval intersect with $\mathbb{Z}_k$ and Distance $\leq d$
5	Count all IDs in $\mathbb{Z}_k$ as c'
6	IF $c' \leq c$, RETURN **false**
7	RETURN **true**

labels, since all matching IDs form one interval of consecutive integers. The look up time for partial keywords can be further accelerated by using additional indexes.

The second index, Neighborhood Interval (NI) index, which is built based on the IDMap index by grouping the labels (IDs) of neighbors of each node into ID intervals. For any node $n_i \in G$, NI index contains ID of n_i, Distance, Label ID interval, Number of indexed neighbor nodes in this entry, and neighbor node IDs. The Distance is the length of the shortest path from n_i to the indexed neighbor node. Given two vertices n_i and n_j in graph G, if there is a directed path from n_i to n_j, n_j is a ***forward neighbor*** of n_i and n_i is a ***backward neighbor*** of n_j.The positive (negative) distance indicates that the indexed node is a forward (backward) neighbor.

There are two pre-defined parameters for the NI index: ***maximum indexed distance*** d_{max} and ***binning factor*** m (m is the maximum number of indexed neighbor nodes in each index entry). The neighbor nodes sharing the same distance are grouped together, ordered by their IDs, and partitioned into rows by the binning factor m. The space required by NI index increases with the increasing the maximum indexed distance d_{max} since more neighbors are indexed for each node. However, larger d_{max} results in better pruning of candidate matches, and improves running time to process connection edges with long distance constraints. NI index is designed to be most effective when partial keywords are specified as prefixes of RDF node labels. NI index can be viewed as a general form of the signatures utilized in both GraphQL [37] and SPath [38]. NI index is also utilized for accelerating the evaluation of connection edges.

3.2 Query Framework

The RDF query framework consists of the following steps:

1. Decomposition of query template into connection edges and components without connections edges.
2. Candidate generation and Pruning: Using IDMap index, generate matching candidates for each query node. Using NI index for Neighborhood Check selectively to prune the candidates.
3. Decomposing each component into a set of smaller basic querying units. We use one level directed trees, named D-trees.
4. Candidate generation for each component: All matching candidates for each D-tree of a component are generated, and joined together to find matching results for each component.
5. Connectivity check: Connection edges between components are processed using NI index to generate the final matches for the query template.

Neighborhood Containment Check (step 2) Component Matching (steps 3 and 4) and Connectivity Check (step 5) are discussed in more detail below.

3.2.1 Neighborhood Containment Check

The neighborhood containment check for NI index is based on ID interval check (here, we assume partial keywords are specified as prefixes of node labels).

The ***ID Interval***, $\mathbb{Z}_j$, of a partial keyword $\wp_j$ of any query node $q_j \in V(G_q)$, is all IDs of node set N_j where $\forall n_i \in N_j$, the label l_i of n_i is a valid match of $\wp_j$.

The ***K-Neighbor*** of a node n_i, where $n_i \in V(G)$, denoted as $Neighbor_k(n_i)$, is a set of nodes that are forward or backward neighbors of n_i via paths of length k or less. That is, if k is positive, $\forall\, n_j \in Neighbor_k(n_i)$, there is a directed path from n_i to n_j with no more than $|k|$ hops; if k is negative, $\forall\, n_j \in Neighbor_k(n_i)$, there is a directed path from n_j to n_i with no more than $|k|$ hops.

Now we define the Neighborhood Check: Given a node $n_i \in V(G)$, and a query node $q_j \in V(G_q)$, n_i passes the ***Neighborhood Check*** of q_j if $\forall\, q_k \in Neighbor_k(q_j)$, ID Interval $\mathbb{Z}_k$ uniquely contains ID of any $n_g \in Neighbor_k(n_i)$, for all $|k| \leq d_{max}$.

An interval of consecutive integers is formed for each partial keyword in query template utilizing IDMap index. Since query templates can contain query nodes with the same partial keyword, value pairs as {Distance, Count (total appearance within Distance)} for each partial keyword are maintained for each query node. The neighborhood check is performed based on partial keywords one by one, and the count of occurrences of this partial keyword is taken into consideration. The term "uniquely contains" in the Neighborhood Check definition means that the node n_g cannot be used to match more than one ID interval. If one partial keyword contains another partial keyword, Count in value pairs is updated. In the **Algorithm** showing the neighborhood check process, {Distance, Count} pairs associated with query node q_j and partial keyword $\wp_i$ are denoted as ψ_j^i. Neighborhood check based on NI index is optimized for partial keywords: 1) only index entries with Label ID interval intersecting with the ID interval of the partial keyword needs to be retrieved; 2) all IDs in the index entries are valid matches for the partial keyword if the ID interval of the partial keyword contains the Label ID interval.

3.2.2 Component Matching

Matching candidates for each component is found by first decomposing the components into D-trees, then joining the matching candidates of the D-trees of the component. Time complexity of component matching is proportional to $\prod_{i=1}^{K} |C_{t_i}|$ where K is number of all decomposed 1 level D-trees and $|C_{t_i}|$ is the number of matching candidates for each D-tree. Finding D-tree decomposition with minimum number of D-trees is likely to improve the time complexity; however it is equivalent to the vertex cover problem [43]. Similar to the vertex cover approximation, we use 2-approximation algorithm to generate D-tree decomposition. Basically, an edge (q_i, q_j) is picked recursively from the query component, and D-trees rooted at q_i and q_j are added to the result. We define selectivity value function $S(q_j) = \frac{deg(q_j)}{|C_{q_j}|}$ which takes both query node's degree and its corresponding candidate set size into consideration as a good measurement of the priority to be

selected as root nodes for two reasons: (i) choosing large degree nodes first is likely to yield better results since D-trees rooted at these nodes can cover more edges in the query component which leads to a smaller K value; (ii) choosing nodes with small candidate sets first is likely to yield less matching candidates for a D-tree. In the second step, NI indexes for all possible root nodes of a decomposed D-tree are checked to generate all D-tree candidate matches. The last step is to join all D-tree candidates together to form component matches. We define the join process as $\boldsymbol{join}(C_i, C_j)$, where C_i and C_j are candidate sets for two subgraphs of G_c (it can either be a decomposed D-tree or joined D-trees). $\boldsymbol{Join}(C_i, C_j)$ combines each pair of matches from two candidate sets by evaluating the predicate: all shared query nodes of two candidate matches need to have equal matching IDs to join. In order to improve the join performance, a new join order J_T for the decomposed D-trees is used as follows: 1. begin with D-tree t_i with smallest candidate set and add t_i to J_T; 2. add D-tree t_j with smallest candidate set to J_T which connects to any already selected D-trees in J_T. Component matching used here is similar to the one used in STWIG [43] with the following differences: 1) D-trees are used as basic join units; 2) new selectivity function is defined based on the size of candidate sets; 3) the NI index is used to generate all D-tree candidates; 4) tree join order is determined by the sizes of tree candidate sets.

3.2.3 Connectivity Check

Connectivity check verifies the paths in the data graph between the nodes that are connected by connection edges in the query graph. If the connection edge is between nodes of the same component then the connectivity check is used to prune the candidates of that component. Otherwise, connectivity check is used to determine whether the two component candidates can join or not. For component connection edges that are within a component, the number of connectivity checks is exactly the size of the component candidate set. For a connection edge between components, the number of connectivity checks depends on the product of the sizes of components' candidate sets. In the worst case, if we have a sequence of N components to be joined by connection edges, the number of connectivity checks that need to be performed can be as large as $\prod_{i=1}^{N} |C_{G_{c_i}}|$. In order to improve query performance, two rules are utilized to determine the order to process connection edges: 1) connection edges inside components are processed before connection edges between components; 2) connection edges between components are processed in the order of the smallest product of candidate sets first. NI index is utilized in processing connection edges. The Connectivity check of connection edge between n_i and n_j is performed by retrieving neighbor IDs of n_i and n_j from NI indexes, and then checks whether the neighbors of n_i intersect with the neighbors of n_j. Here, we assume the maximum indexed distance d_{max} of neighborhood index is greater than $Ceil(\frac{d_c}{2})$, where d_c is the distance specified with connection edge. Otherwise, we need to combine index entries of n_i and n_i's neighbor nodes together in order to get more hops of neighborhood information for n_i.

4. DATASETS

The UniProt Knowledgebase (UniProtKB) [25] is a central hub of protein information which provides an integrated view of association and interaction data from different biomedical datasets. One important motivation of UniProtKB is to allow users query the related but dispersed information across disparate protein related datasets. Each protein entry recorded in UniProtKB provides a variety of information related to this protein including protein and gene names (mnemonic name, structured name and alternate names), protein sequences, protein function, catalytic activity, co-factors, subcellular localization, patterns of expression, protein–protein interactions, and disease association. Besides the rich information provided for each protein, another advantage of UniProtKB is its high update rate and availability in different formats. UniProtKB data is released every 4 weeks to provide the most up to date protein information in multiple formats including plain text, XML, RDF and GFF.

The decisive factor to choose UniProtKB rather than another database available in RDF format for our experiment relies on its high quality and accuracy of data integration. Since RDF format has no pre-defined schema, RDF data is designed to integrate with ease by combining the triples from different sources directly if unified resource identifiers are utilized. For UniProtKB, each entry undergoes both automated and manual checks to ensure the high accuracy and consistency of the data before it is integrated. The automated check is performed through a quality control software to ensure the correctness of syntax and verification of different biological rules for the entry. Besides this, the manual review process provides extra effort to ensure that all relevant literature, annotation and analysis results are included. As the correctness of the querying results across multiple datasets is determined by the lowest quality data integrated, the high quality standards provided by UniProtKB is essential to provide high confidence of querying results in the experiments.

4.1 Integrated Datasets

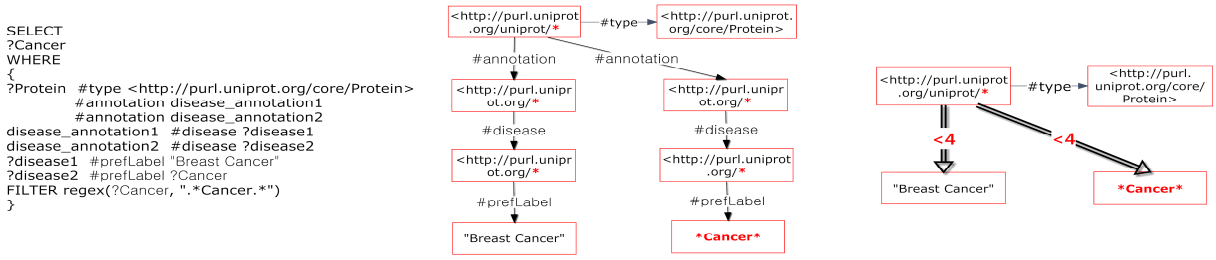
In this section, we describe the three integrated datasets in UniProtKB which are extracted and queried in our experiments reported in the next section: IntAct (protein-protein interaction), Reactome (pathway), and OMIM (disease and phenotype).

4.1.1 IntAct

IntAct [26] provides open-source molecular interaction data populated by interactions curated from the literature, as well as from direct data depositions. The information within the IntAct database primarily consists of protein–protein interaction (PPI) data. An important aspect of the IntAct dataset is that each entry in IntAct is peer reviewed by a senior curator, and not released until accepted by that curator. UniProtKB database is readily integrated with the IntAct database to provide protein–protein interaction data. In order to meet the required quality standard of UniProtKB, only a subset of high quality interactions are imported from IntAct based on a statistical scoring system. A score threshold is chosen by UniProtKB to exclude binary interactions supported by only one experimental observation. In addition to the score-based filter, a set of defined rules are utilized to exclude certain types of data, such as interactions observed in larger complexes, or interactions that have not been experimentally validated. By using these strict criteria, only experimentally validated binary interactions supported by multiple observations are imported into UniProtKB.

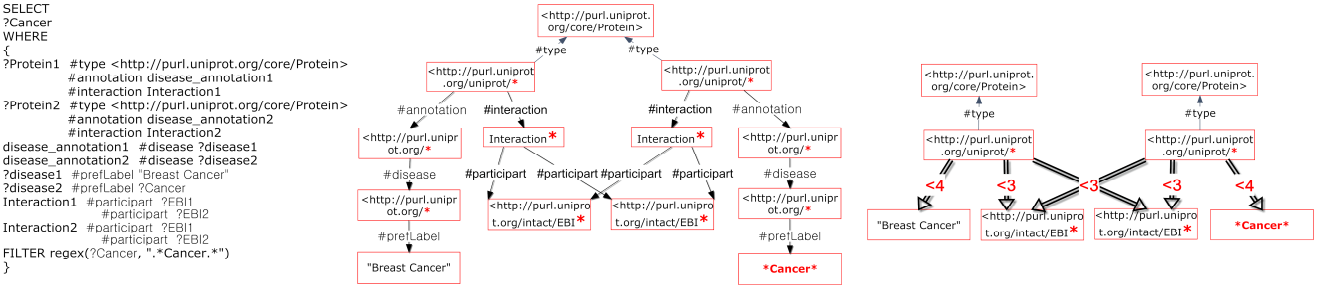
4.1.2 Reactome

Reactome [27] is a manually curated open-source human pathway and reaction dataset. In order to provide a unified identifier, Reactome merges pathway identifier mapping, over-representation and expression analysis tools into a single portal. Reactome uses UniProtKB protein identifiers to provide a list of pathways in which the protein functions. Compared with the pathway annotation provided by UniProtKB directly, the cross referenced Reactome pathways provide more complete information for each protein.



(a) SPARQL (b) Complete Query Template (c) GBE Query Template

Figure 3. Query Specifications of Q3



(a) SPARQL (b) Complete Query Template (c) GBE Query Template

Figure 4. Query Specifications of Q5

4.1.3 OMIM

Online Mendelian Inheritance in Man (OMIM) database [28] is utilized in UniProtKB to provide disease/phenotype information for disease annotations associated with proteins. OMIM is a comprehensive, authoritative and timely knowledgebase of human genes and genetic disorders. Each OMIM entry has a full-text summary of a genetically determined phenotype. UniProtKB carefully links the OMIM entry with the protein entry and describes the natural variant(s) of the protein sequence potentially associated with disease according to the scientific literature.

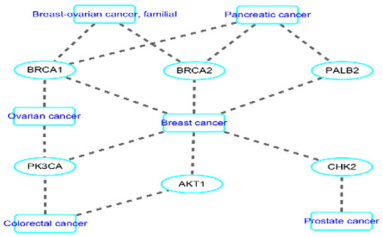
4.2 Data Extraction

The data utilized in our experiments is a subset of triples from the UniProtKB RDF dataset. We focus on only human proteins that currently have active entries. The UniProtKB raw data, downloaded from the UniProt website on 4-10-2015, contains about 150 million triples from 971,583 (both reviewed and unreviewed) protein entries. Note that different protein isoforms are represented as different protein entries in UniProtKB. In order to provide a more concise RDF graph to support efficient signature-based indexes, only protein entries associated with at least one of the following statements are extracted: disease annotation, function annotation, PTM annotation, cofactor annotation, subunit annotation and protein interaction. For each protein entry, the following properties are extracted: protein names (mnemonic name, recommended name and alternate name), protein organisms (including the taxonomy information), protein keywords, protein tissues, protein gene information (including different gene labels), and protein pathway information (Reactome pathway associated with the protein). The extracted RDF graph contains 89,915 proteins (including different protein isoforms), 4,211 diseases, 1,278 pathways, 18,243 interactions and 35,063 annotations. The size of the extracted RDF graph is about 11.6 million triples including 9 million triples from the taxonomy data. Data extraction can be systematically done from any version of the UniProtKB RDF graph, and more types of annotations can be extracted by changing the specification of the data extraction process.

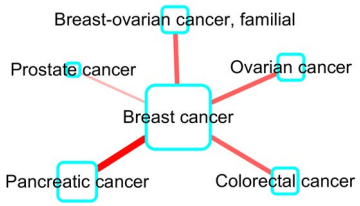
5. EXPERIMENTAL RESULTS

The proposed framework is implemented with Visual C# 2010 and SQL Server 2008. All experiments were performed on a 2.93GHZ Intel(R) Xeon machine with 48GB RAM running Windows Server 2008 R2. The average space needed for the NI index is $O(N(\mu/2)^{d_{max}}/m)$, where N is the number of vertices in G, μ is the average node degree, d_{max} is the maximum hops of neighbors indexed and m is the binning factor. The NI index with a larger d_{max} value results in higher pruning power and ability to handle connection edges with large distance constraints at the cost of requiring more storage space. By using the IDMap index to hash the RDF labels into IDs, the 2 hop NI index achieves a similar size as the original RDF graph while 3 hop NI index is 8 times larger. As we explain in Section 3.2, 2 hops NI index can evaluate connection edges with distance constraints up to 4 hops efficiently which is sufficient for all proposed queries. Here, we decide to use 2 hops NI indexes for graph template matching.

In this section, we primarily investigate the ease of utilization of GBE query templates to specify integrated queries and the strength of these queries in discovering interesting patterns across the integrated network from disparate biological entities. For this purpose, we focus on 10 queries that require integrated querying across multiple interaction and/or association datasets. We categorize these 10 queries into two groups: 1) single protein patterns: querying the relationships among one protein and other types of resources; 2) multiple protein patterns: querying protein-protein joining based on protein-protein interactions, shared pathways, shared diseases, or shared function. Rather than displaying the results in tabular form, we export query results into Cytoscape [29] to produce meaningfully summarized graphs. For each query, the query results can produce multiple summarized graphs by specifying different relationships among various types of resources.



(a) Disease-Protein



(b) Disease-Disease

Figure 5. Query Result Visualization of Q3

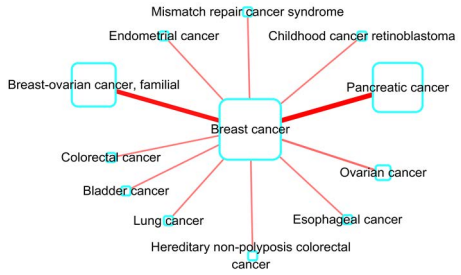


Figure 6. Disease-Disease Relationships for Q5

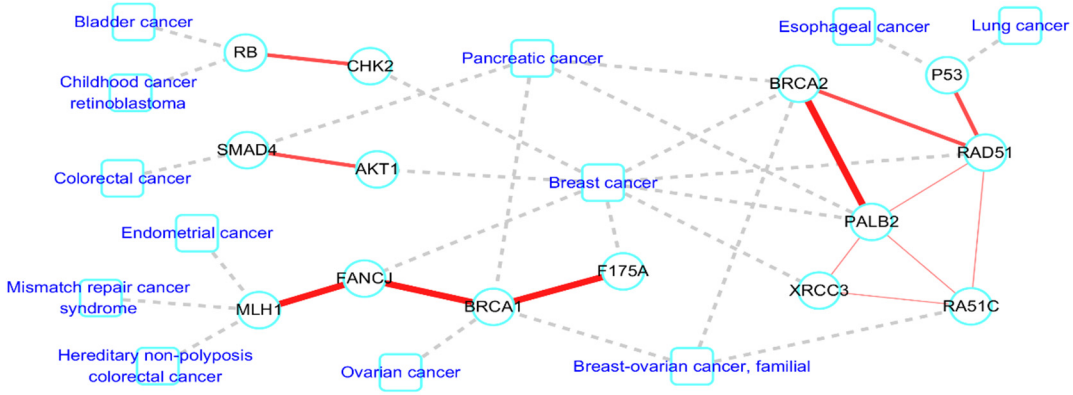


Figure 7. Full Visualization of Query Results for Q5

5.1 Single Protein Patterns

Single protein patterns focus on querying relationships between one protein and other types of resources across different datasets. Four queries are proposed as single protein patterns:

Q1. Finding pathways that contain at least one protein associated with "Breast Cancer".

Q2. Finding molecular functions that are associated with at least one protein associated with "Breast Cancer".

Q3. Finding cancers that are associated with at least one protein that is also associated with "Breast Cancer". (Motivated by Figure 1 (a))

Q4. Finding tissues that are associated with at least one protein which is associated with "Breast Cancer".

To evaluate these four queries, at least two types of biological resources/entities need to be integrated: protein resource (Uniprot) and disease resource (OMIM). UniProtKB has already linked protein resource with disease information which makes some of queries solvable through manual effort, e.g. one can search all protein entries associated "Breast Cancer" on Uniprot website and manually check all these entries to find any other cancers associated. As UniProtKB also provides a beta SPARQL endpoint which allows user to specify SPARQL queries, some of these queries can also be specified in SPARQL. However, both manual check and specifying the complete SPARQL queries requires significantly more effort compared with using GBE query templates in our framework.

In Figure 3, we illustrate three possible ways to specify Q3. One can easily observe that both alternatives of specifying the complete query template (Figure 3 (b)) and the SPARQL query (Figure 3 (a)) need not only strong database background but also a good understanding of the underlying graph structure of the UniProt RDF graph. In comparison, by using connection edges and partial keywords, GBE query templates can be much simpler to formulate. As shown in Figure 3 (c), the GBE query template uses connection edges to avoid specifying the intermediate annotation node and the disease ID node in identifying the relationship between a protein and disease name. With a little background of UniProt RDF dataset, the GBE query framework provides simpler query specifications as demonstrated by the abstract queries in Figure 1. Note that, using connection edges may yield more results compared to specifying all edges in the complete graph template. For all these four queries, the simplified GBE query templates return the same set of results as the complete graph template on the extracted UniProtKB graph. The GBE framework also supports displaying the results in graph templates. Thus, the researchers can choose a subset of the results they are interested, and display them as graph templates to perform exploratory search. Due to space limits, the details of using the GBE framework to display query results as templates to perform exploratory search is not included here.

Q1 returns 60 Reactome pathways potentially associated with "Breast Cancer". 13 protein function annotations associated with proteins which are associated with "Breast Cancer" are found in Q2. Q3 discovers 5 cancers that have overlapping molecular bases with "Breast Cancer". 23 tissues are identified in Q4 related to proteins associated with "Breast Cancer". As we explain in Section 2, identification of diseases with overlapping molecular bases may be useful in identifying unknown disease-disease relationships. Here, we display the results of Q3 in Figure 5 using Cytoscape. In Figure 5 (a), we display the connections among proteins and diseases identified in the query results of Q3 by making the disease and the protein in each query result as the source and target in Cytoscape. To further provide more clear relationships between "Breast Cancer" and other cancers, another summarized graph is produced in Figure 5 (b) by making the two diseases in each query result as the source

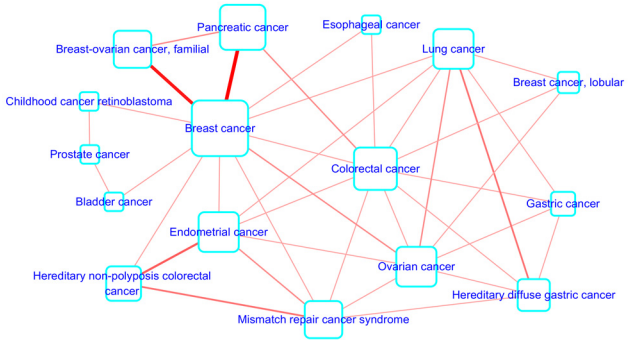


Figure 8. Disease-Disease Relationships for Q6

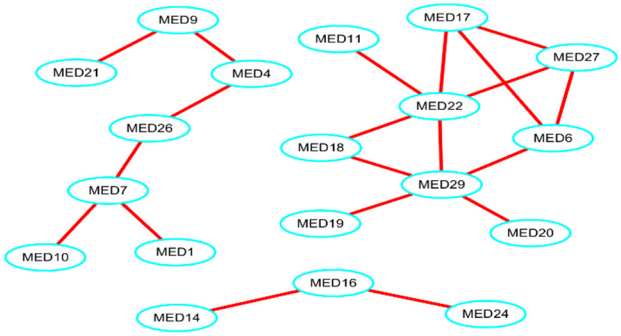


Figure 9. Protein-Protein Relationships for Q7

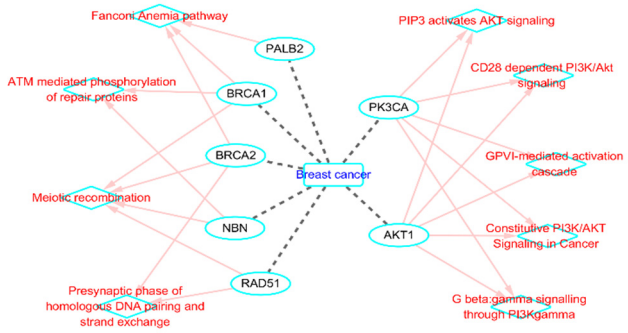


Figure 10. Protein-Protein Relationships for Q8

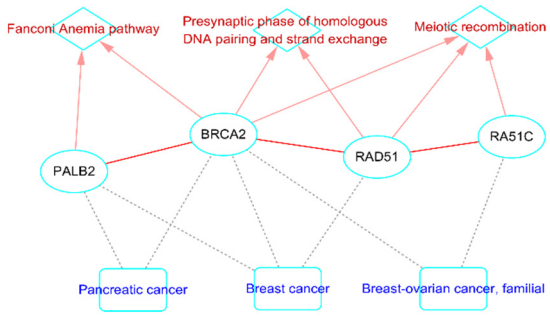


Figure 11. Protein-Protein Relationships for Q10

and target in Cytoscape. The width of the edge between two diseases is determined by the number of proteins shared by this disease pair. The size of the cancer node is based on the number of shared proteins. The more proteins shared between one disease and "Breast Cancer", the larger disease node is (same for edge width, the more shared proteins, the wider the edge is). One can observe that "Pancreatic Cancer" and "Breast Cancer" have strong shared molecular bases (3 shared proteins) and are more likely to relate with each other. Such a network is utilized in [7] which can be constructed via a single query in our framework.

5.2 Multiple Protein Patterns

Compared to single protein patterns, multiple protein patterns are more complex for query specification and evaluation. We consider three types of protein-protein joining conditions: protein-protein interaction, shared functions and shared pathways. Some of the multiple protein pattern queries also require joining proteins based on multiple criteria.

5.2.1 Protein-Protein Interaction

Protein–protein interactions reveal functional relationships between genes. Though a large number of protein–protein interaction datasets are available, no direct way is proposed for querying patterns of protein–protein interactions related with other types of biological entities, e.g. certain phenotypes. By using the extracted UniprotKB graph, we demonstrate the potential power of seamlessly querying the integrated biology datasets. We consider two queries here:

Q5. Finding cancers associated with at least one protein that interacts with another protein associated with "Breast Cancer". (Motivated by Figure 1 (b))

Q6. Finding all pairs of cancers associated with pairs of proteins that interact with each other. (Motivated by Figure 1 (b))

Specifying queries of multiple protein patterns is more difficult compared to queries involving single protein patterns. Similar to Figure 3 of Q3, all three possible ways (SPARQL, complete graph template and simplified GBE template) to specify Q5 are shown in Figure 4. Note that the binary interaction between two proteins are identified by two EBI observations which is the requirement of UniprotKB dataset.

Q6 is a more general form of Q5 as Q6 tries to identify all pairs of cancers with shared molecular interactions while Q5 specifies one disease as "Breast Cancer". Complete visualization of query results for Q5 is shown in Figure 7. The width of the edge between two proteins (represents the binary interaction between two proteins) is proportional to the number of disease pairs shared this interaction. Four protein-protein interaction patterns are identified to connect different cancers with "Breast cancer". Compared with Figure 5 (a), one can observe that there is no edge between BRCA1 protein and "Breast Cancer" in Figure 7. This is a result of graph isomorphism matching of the query template as each query node should match a unique node in the RDF graph. Since there is no protein associated with other cancers which interacts with BRCA1 protein, no query results contain an edge between BRCA1 protein and "Breast Cancer" for Q5. The summarized disease-disease relationships identified by Q5 is shown in Figure 6. Similar to Figure 5 (b), the edge width represents the number of protein-protein interactions shared by the respective disease pair. For Q6, the complete disease-disease relationships among all pairs of cancers is shown in Figure 8. These two queries are motivated by Section 2, Figure 1 (b) as finding shared molecular interactions between diseases.

5.2.2 Shared Functions

Identifying network schemas requires protein-protein joining through both shared functions and protein-protein interactions as shown in Section 2, Figure 1 (c). We consider one query:

Q7. Finding pairs of proteins which have general function description as "Component of the Mediator complex" and interact with each other. (Motivated by Figure 1 (c))

The query results are displayed in Figure 9 using Cytoscape. Three groups of proteins are identified where proteins closely interact with each other in the same group.

5.2.3 Shared Pathways

Understanding the roles of proteins in higher order interconnected pathways is critical to understanding molecular reactions at cellular level. Some conceptualizations of protein-protein interactions also consider proteins in the same pathway interact with each other directly or indirectly. Identifying protein-protein patterns with shared pathways provides complementary information to binary protein-protein interactions. Here, we consider three queries:

Q8. Finding any pair of proteins that are involved in the same pathway and are both associated with "Breast Cancer".

Q9. Finding cancers associated with a protein that is involved in the same pathway with a protein associated with "Breast Cancer". (Motivated by Figure 1 (e))

Q10. Finding two proteins that are involved in the same pathway and interact with each other such that one protein is associated with "Breast Cancer" and the other is associated with another cancer.

The query results of Q8 are visualized in Figure 10. We observe that there are two groups of proteins where the proteins in each group are closely related with each other through shared pathways. Four proteins associated with three cancers which interact with each other and share three common pathways are identified in Q10 as shown in Figure 11. Note that "Breast Cancer", "Pancreatic Cancer" and "Breast-Ovarian Cancer" are also observed to share molecular interactions in Figure 6 and Figure 8. The query results of Q9 returns 24 distinct proteins, 31 pathways and 15 cancers (not include "Breast Cancer"). Q9 is motivated by Section 2, Figure 1 (e) as finding diseases with shared pathways.

5.3 Query Evaluation Efficiency

The neighborhood signature index is considered as a strong candidate to evaluate GBE templates as it can: 1) reduce the unnecessary candidates in subgraph isomorphism matching, and 2) handle connection edges with short distance constraints efficiently. By indexing all neighbor node IDs in the NI index, the connection edges can be evaluated efficiently. Query running time and the number of matching results for all 10 queries are shown in Table 1. We find that single protein pattern queries are more efficient to evaluate compared with multiple protein pattern queries. For single protein pattern query, Q2 requires the longest running time as the component of matching any protein associated with function annotations results in a large number of matching candidates which leads to a large number of connection checks. Q6 is the most time consuming multiple protein pattern query as each decomposed component is small and all components are connected through connection edges. Compared with Q6, Q5 requires less effort as one of the cancers is specified as "Breast cancer" which significantly reduce the intermediate matches. Except for Q6, all queries can be evaluated in less than thirty seconds or so which is good enough to support real time biomedical applications.

Table 1. Query Running Times for all Queries

Query ID	Q1	Q2	Q3	Q4	Q5
# of Results	72	13	10	48	25
Run Time	3.86 *s*	10.26 *s*	1.42 *s*	1.06 *s*	23.31 *s*
Query ID	Q6	Q7	Q8	Q9	Q10
# of Results	67	20	32	158	8
Run Time	156.4 *s*	15.71 *s*	15.79 *s*	22.09 *s*	31.74 *s*

6. RELATED WORK

RDF systems including Sesame [31], Virtuoso [32], 3Store [33] and Jena [34], use SPARQL [30] as default query language. SPARQL uses graph patterns as basic query units and is evaluated by triple joins. Currently, SPARQL extends its functionalities to support paths defined with regular expressions (*property paths*). However, most of these systems have difficulties in answering the GBE templates similar to the examples shown in Figure 3(c) and 4(c): 1). connection edges are complex to answer without specific indexes; 2). partially entered labels are not properly addressed; 3). query templates may generate too many intermediate candidates if no pruning techniques are utilized.

Signature based indexes are commonly utilized in literature to accelerate graph isomorphism matching [35-40]. TALE [35] and SAPPER [36] use similar techniques where the labels of neighbors for each node are indexed by hashing into bit arrays and these bit arrays are utilized as signatures to check the neighborhood containment. GraphQL [37] and SPath [38], on the other hand, use neighborhood indexes directly index node labels. In GraphQL, the neighbors of each node are indexed as a sequence of node labels in lexicographic order. gStore [39, 40] extends the utilization of signature-based pruning to RDF datasets. The neighborhood signature is proposed as a bit string according to the adjacent edge labels and node labels. All vertex neighborhood signatures are then indexed using a special index schema, VS tree, to provide efficient query evaluation.

Storing biology data in RDF is a current move. In addition to the UniProt dataset utilized in our experiments, many other biology datasets are available in RDF format from the Bio2RDF project [41], including chEMBL, ClinicalTrials, DrugBank, iProClass, KEGG, MeSH, Pharmacogenomics Knowledge Base.

7. CONCLUSIONS

In this paper we have demonstrated that, using simple graph templates to query an integrated RDF knowledge base, graph template matching based querying is a promising tool to express sophisticated questions, and discover hidden connections among biological or biomedical entities from diverse datasets. For data to be published or exchanged in the web, RDF is increasingly being adopted, and vast amounts of RDF data are already available. The availability of ever increasing amounts of RDF data in various fields of biomedical applications from public and private sources, adds even more to the potential of querying integrated RDF data sets for new insights and discovering knowledge about hidden associations among biological and medical entities.

Future work includes building a web-based graphical user interface to enable biomedical researchers to use this framework for exploratory querying. Using more diverse and much larger collections of RDF data, including data on publications, sequences, and drugs as data sets, will enable us to query even more sophisticated queries by directly using graph template querying. Finally, we are currently working on the distributed version of our querying framework and combining our work with compressive genomics data.

8. REFERENCES

[1] Mosca, R., Pons, T., Céol, A., Valencia, A., and Aloy, P. 2013. Towards a detailed atlas of protein–protein interactions. Current opinion in structural biology, 929-940.

[2] Carter, S. L., Brechbühler, C. M., Griffin, M., and Bond, A. T. 2004. Gene co-expression network topology provides a framework for molecular characterization of cellular state. Bioinformatics, 20(14), 2242-2250.

[3] Hu, Z., Killion, P. J., and Iyer, V. R. 2007. Genetic reconstruction of a functional transcriptional regulatory network. Nature Genetics, 39(5), 683-687.

[4] Pah, A. R., Guimera, R., Mustoe, A. M., and Amaral, L. A. N. 2013. Use of a global metabolic network to curate organismal metabolic networks. Scientific reports, 3.

[5] Tong, A. H. Y., Lesage, G., Bader, G. D., Ding, H., Xu, H., Xin, X... 2004. Global mapping of the yeast genetic interaction network. Science, 303(5659), 808-813.

[6] Ritz, A., Tegge, A. N., Kim, H... 2014. Signaling hypergraphs. Trends in biotechnology, 32(7), 356-362.

[7] Goh, K. I., Cusick, M. E., Valle, D., Childs, B., Vidal, M., Barabási, A. L. 2007. The human disease network. PNAS, 104(21), 8685-8690.

[8] Von Eichborn, J., Murgueitio, M. S., Dunkel, M... 2011. PROMISCUOUS: a database for network-based drug-repositioning. Nucleic acids research, 39. D1060-D1066.

[9] Vogt, I., and Mestres, J. 2010. Drug-target networks. Molecular Informatics, 29(1-2), 10-14.

[10] Lisewski, A. M., Quiros, J. P... 2014. Supergenomic network compression and the discovery of EXP1 as a glutathione transferase inhibited by artesunate. Cell, 158(4), 916-928.

[11] Li, Y., and Patra, J. C. 2010. Genome-wide inferring gene–phenotype relationship by walking on the heterogeneous network. Bioinformatics, 26(9), 1219-1224.

[12] Cheng, F., Liu, C., Jiang, J... 2012. Prediction of drug-target interactions and drug repositioning via network-based inference. PLoS computational biology, 8(5), e1002503.

[13] Cao, M., Pietras, C. M., Feng, X., Doroschak, K. J... 2014. New directions for diffusion-based network prediction of protein function: incorporating pathways with confidence. Bioinformatics, 30(12), i219-i227.

[14] Liu, Y., Koyutürk, M... 2012. Integrative analysis of common neurodegenerative diseases using gene association, interaction networks and mRNA expression data. AMIA Summits Transl Sci Proc, 2012, 62.

[15] Menche, J., Sharma, A... 2015. Uncovering disease-disease relationships through the incomplete interactome. Science, 347(6224), 1257601.

[16] Li, J., Zheng, S., Chen, B., Butte, A. J., Swamidass, S. J., and Lu, Z. 2015. A survey of current trends in computational drug repositioning. Briefings in bioinformatics, bbv020.

[17] M. Koyuturk. Using protein interaction networks to understand complex diseases, IEEE Computer, 31-38, 2012.

[18] Banks, E., Nabieva... 2008. NetGrep: fast network schema searches in interactomes. Genome biology, 9(9), R138.

[19] Pandey, J., Koyutürk, M., Kim, Y., Szpankowski, W... 2007. Functional annotation of regulatory pathways. Bioinformatics, 23(13), i377-i386.

[20] Z. Kaoudi, Ioana Manolescu. RDF in the Clouds: A Survey. VLDB Jounal, Springer-Verlag, 2014.

[21] Baryshnikova, A., Costanzo, M., Dixon, S., Vizeacoumar, F. J... 2010. Synthetic genetic array (SGA) analysis in Saccharomyces cerevisiae and Schizosaccharomyces pombe. Methods in enzymology, 470, 145-179.

[22] Vinayagam, A., Zirin, J... 2014. Integrating protein-protein interaction networks with phenotypes reveals signs of interactions. Nature methods, 11(1), 94-99.

[23] Falck, J., Petrini, J. H... 2002. The DNA damage-dependent intra–S phase checkpoint is regulated by parallel pathways. Nature genetics, 30(3), 290-294.

[24] Bailly-Bechet, M., Borgs... Finding undetected protein associations in cell signaling by belief propagation. PNAS, 108(2), 882-887.

[25] UniProt C. 2015. UniProt: a hub for protein information. Nucleic Acids Res, 43: D204–12.

[26] Kerrien S, Aranda B, Breuza L, Bridge A, Broackes-Carter F, Chen C... 2012.The IntAct molecular interaction database. Nucleic Acids Res. 2012

[27] Croft D, Mundo AF, Haw R, Milacic M, Weiser J, Wu G, et al. 2014. The Reactome pathway knowledgebase. Nucleic Acids Res. 42(D1):D472–7.

[28] Hamosh A., Scott A.F... 2002. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. Nucleic Acids Res.

[29] Lopes C.T., Franz M., Kazi F., Donaldson S.L., Morris Q., Bader G.D. Cytoscape Web: an interactive web-based network browser. Bioinformatics 2010.

[30] J. P´erez, M. Arenas, and C. Gutierrez. Semantics and Complexity of SPARQL. ACM TODs, 2009.

[31] J. Broekstra, A. Kampman, and F. Harmelen. Sesame: A generic architecture for storing and querying RDF and RDF Schema. Proc. of ISWC, pages 54-68, 2002.

[32] O. Erling and I. Mikhailov. RDF support in the virtuoso dbms. In CSSW, pages 59–68, 2007.

[33] S. Harris and N. Gibbins, 3store: Efficient bulk RDF storage. Proc. of PSSS, pages 1-15, 2003.

[34] K. Wilknson, C. Sayers, H. Kuno... Efficient RDF storage and retrieval in Jena2. PSWDB, pages 131-150, 2003.

[35] Tian, Y., & Patel, J. M. Tale: A tool for approximate large graph matching. ICDE 2008, 963-972 (2008).

[36] Zhang, S., Yang, J., & Jin, W. (2010). SAPPER: subgraph indexing and approximate matching in large graphs. PVLDB, 3(1-2), 1185-1194 (2010).

[37] He, H., & Singh, A. K. Graphs-at-a-time: query language and access methods for graph databases. SIGMOD, 405-418 (2008).

[38] Zhao, P., & Han, J. On graph query optimization in large networks. PVLDB, 3(1-2), 340-351 (2010).

[39] Zou, L., Ozsu, M. T., Chen, L., Shen, X., Huang, R., & Zhao, D. gStore: a graph-based SPARQL query engine. VLDB Journal, 23(4), 565-590 (2014).

[40] Zou, L., Mo, J., Chen, L., Ozsu, M. T., & Zhao, D. gStore: answering SPARQL queries via subgraph matching. PVLDB, 4(8), 482-493 (2011).

[41] F. Belleau, M. Nolin, N. Tourigny, P. Rigault and J. Morissette. Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. 2008. Journal of Biomedical Informatics 41:5 p706-716.

[42] Shi Qiao, Z. Meral Ozsoyoglu. One Size Does not Fit All: When to Use Signature-based Pruning to Improve Template Matching for RDF graphs. arXiv:1501.07184.

[43] Sun, Z., Wang, H., Wang, H., Shao, B., & Li, J. 2012. Efficient subgraph matching on billion node graphs. PVLDB, 5(9), 788-799.