BIOINFORMATICS ORIGINAL PAPER Vol. 24 no. 7 2008, pages 1002–1010
doi:10.1093/bioinformatics/btn054

Databases and ontologies

Statistical search on the Semantic Web

Norio Kobayashi and Tetsuro Toyoda*

Integrative Omics Research Team, Computational and Experimental Systems Biology Group, Genomic Sciences Center, RIKEN, 1-7-22 Suehiro, Tsurumi, Yokohama, Kanagawa 230-0045, Japan

Received on August 22, 2007; revised on February 3, 2007; accepted on February 6, 2007

Advance Access publication February 8, 2008

Associate Editor: Alex Bateman

ABSTRACT

Motivation: Statistical analysis of links on the Semantic Web is important for various evaluation purposes such as quantifying an individual's scientific research output based on citation links. SPARQL has been proposed as a standardized query language for the Semantic Web and is intuitively understandable; however, it does not adequately support statistical evaluation of semantic links.

Results: We have extended SPARQL to a novel Resource Description Framework (RDF) query language termed General and Rapid Association Study Query Language (GRASQL) to generate inferences connecting semantic Boolean-based deduction and statistical evaluation of RDF resources. We have verified the descriptive capability of GRASQL by writing GRASQL queries for practical biomedical search patterns including *in silico* positional cloning studies and for ranking researchers in a specific domain of expertise by introducing *k* index, the number of papers containing specific keywords that are published in a fixed period by a researcher. We have also developed a search engine termed General and Rapid Association Study Engine (GRASE), which executes a restricted variety of GRASQL queries by requesting a dynamic and comprehensive evaluation of statistical significance of intersections between each group of documents assigned to URIs and those documents matching user-specified keywords and omics conditions. By performing practical *in silico* positional cloning searches with GRASE, we show the relevance of our approach on the Semantic Web for biomedical knowledge discovery problem solving.

Availability: GRASE is used as the search engine for the Positional Medline (PosMed) service and Researcher Finder service at http://omicspace.riken.jp/

Contact: toyop@gsc.riken.jp

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

The Semantic Web is a framework for knowledge description and discovery by inferences, which uses relationships given as semantic links between two entities denoted by URIs (Berners-Lee *et al*., 2001). A goal of the Semantic Web is the realisation of communication between humans and machines by adding metadata describing the semantic links of entities based on Resource Description Framework RDF; (Manola and Miller, 2004). In biomedical fields, some datasets using common ontology shared by people on the Semantic Web such as Gene Ontology (Ashburner *et al*., 2000) and uniprot RDF (http://dev.isb-sib.ch/projects/uniprot-rdf/) have been published in RDF. However, because the task of generating consistent RDF triples (subject, predicate and object) against a vast number of biomedical contents is too expensive, an information space that covers our entire exhaustive biomedical knowledge on the Semantic Web has not been realized.

Several languages for querying RDF triples, such as ARQ (Seaborne, 2006), Oracle-RDF (Chong *et al*., 2005), SPARQL (Prud'Hommeaux and Seaborne 2008), RDQL (Seaborne, 2004) and RQL (Karvounarakis *et al*., 2003), have been proposed thus far. Although the expressiveness of each language is different, they are basically designed for simple graph matching against RDF triples; a graph pattern of RDF triples is described as a query in a language and the answer is obtained by pattern matching against a set of RDF triples. Among those, SPARQL is highly evaluated, because it seems intuitively understandable for typical biologists who are not familiar with programming languages, although it requires statistical analyses. However, SPARQL does not adequately support statistical evaluation of semantic links. Our need is a SPARQL-based statistical querying that applies statistical methods to sets of RDF triples obtained by RDF graph pattern matching. For instance, we would like to rank resultant entities by considering statistical values computed for each entity based on sets of RDF triples hit by pattern matching, in order to discover entities statistically associated with a user's keyword.

For the practical use of published biomedical data on the Semantic Web, it is beneficial to reinforce the acquisition of valuable data that is difficult to utilize because of the lack of semantic links by supplying a hybrid methodology combining not only inferences over the knowledge described with RDF but also those supported by statistical significance over multiple raw documents. For instance, MEDLINE, a biomedical document repository, includes over 16 million reports; the entire knowledge of MEDLINE cannot be consistently reconstructed as well-formed knowledge based on ontology.

In much biomedical research, an initial discovery that two entities may have a certain relationship is reported first, then their detailed relationship is discussed, analyzed and verified

*To whom correspondence should be addressed.

extensively in additional reports. It is sometimes effective to investigate the strength of the relationship between two entities by statistically evaluating their co-citation frequencies in a large number of reports, rather than to wait until the final definition of the relationship is determined (Jenssen *et al.*, 2001). Furthermore, the World Wide Web, the web for short, is a widely spread powerful technology that realizes a global information space of data and services. On the web, the evaluation of significance of each document by statistical analysis of the topology of documents connected with hyper links is effectively performed in document search engines such as Google (Page *et al.*, 1998).

However, the current framework of the Semantic Web cannot handle numerical criteria such as strength of relationship and a statistical test of the relationship. That is, in order to effectively utilize both well-formed RDF datasets and a vast number of biomedical documents, the extension of current query languages should support not only Boolean relationships but also statistically evaluated relationships. We have defined a novel query language termed General and Rapid Association Study Query Language (GRASQL), by introducing procedures associating entities based on the statistical measurements to SPARQL, which is a query language standardized in World Wide Web Consortium (W3C) for the Semantic Web. We have confirmed that it has the capability to describe queries for several practical problems in the biomedical field.

Furthermore, we have developed a prototype version of a rapid search engine termed General and Rapid Association Study Engine (GRASE), which substantially supports the search logic written in GRASQL; however, it does not fully support GRASQL. Using the prototype of the search engine in this article, we evaluate the benefits of utilizing statistical inference in addition to the explicit definition of semantic relations.

2 METHODS

2.1 Motivating examples

We start with a researcher ranking problem as our motivating example to show how statistical evaluation is integrated with existing RDF search.

The first example is researcher ranking using an index to characterize the scientific output of a researcher called *h* index (Hirsch, 2005). The *h* index is introduced as follows: 'A scientist has index *h* if *h* of his or her N_p papers have at least *h* citations each and the other $(N_p - h)$ papers have at most *h* citations each', where N_p is the number of papers published over *n* years. Figure 1 shows a GRASQL query of this problem, which uses MEDLINE abstracts and citation relationships. First, we obtain a set $\bar{d}_r$ of documents published by each researcher *r* over the MEDLINE abstracts published in 2003 or later, then documents $\bar{c}_d$ that cite document $d \in \bar{d}_r$ each. This search implements RDF graph pattern matching specified in the WHERE clause in Figure 1, as shown in Figure 2. Then, we compute *h* index for each researcher. This statistical step cannot be realized with RDF graph pattern matching unless an external procedure against the sequences of solutions obtained in the first step is used. In Figure 1, the EVALUATE clause, which is newly introduced in GRASQL, specifies a computation method of *h* index `?h` for each researcher `?researcher` using the external statistical function `ris:hIndex` given in Supplement 1. Since MEDLINE does not contain citation information, the `rip:hasCitation` links should be generated based on other resources, such as Google Scholar (Noruzi, 2005).

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate/>
@prefix ris: <http://omicspace.riken.jp/GRASQL/statistics/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@let  %documentSet  rio:MEDLINE
@let  %researcher  rio:Researcher
SELECT ?researcher ?h
WHERE {
  ?researcher  rip:hasDocument  ?doc ;
               rdf:type         %researcher .
  ?docCite  rip:hasCitation   ?doc ;
            rdf:type          %documentSet .
  ?doc  rip:publishYear   ?year ;
        rdf:type          %documentSet .
  FILTER (?year >= 2003)
}
EVALUATE ?h FOR ?researcher {
  ?h = ris:hIndex([?doc,?docCite])
}
ORDER BY DESC(?h)
```

Fig. 1. A GRASQL query that ranks researchers by *h* index using MEDLINE abstracts and citation relationships. The statistical function `ris:hIndex` in the EVALUATE clause is called for each `?researcher` containing the sequences of solutions obtained by RDF graph pattern matching specified in the WHERE clause. `[?doc,?docCite]` is a sequence of pairs of `?doc` and `?docCite` included in the sub-sequences of solutions concerning the value of `?researcher` when `ris:hIndex}` is called.

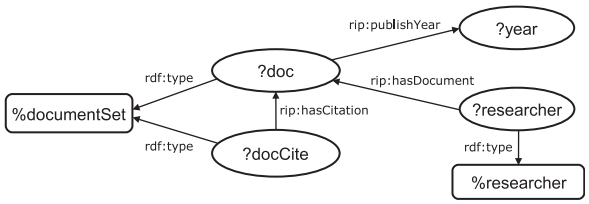

Fig. 2. RDF graph pattern specified in the WHERE clause in Figure 1.

The second example is for ranking researchers in a topic specified with a keyword. That is, we would like to rank researchers by considering $N_{r,k}$ number of documents written by a researcher *r* including a keyword *k*. We call the number $N_{r,k}$, *k* index of researcher *r*. Figure 3 shows a query for this example in GRASQL. The documents `?doc` written by the researcher `?r` including the keyword `%keyword` is obtained not only by RDF graph pattern matching but also by calling an external program specified in the WHERE clause shown in Figure 4. The predicate `rix:hasWord` of this example is used to call a full-text search engine for finding MEDLINE abstracts including the keyword and the results are cached in RDF graphs. Further, *k* index, namely the number of documents `?doc` without duplication for each researcher `?r`, is computed by calling the statistical function `ris:countDistinct` in the EVALUATE clause. An execution result of this example is shown in Section 3.2.3.

2.2 Design of a statistical RDF query language

At present, concerning the realization of statistical computation against RDF triples, some existing RDF query languages that support external procedure calls can describe such a statistical query. For instance, in Oracle-RDF, statistical procedures can be implemented as stored procedures and applied to a result of the `RDF_MATCH` table function that achieves querying RDF data. Furthermore, as an RDF query language extended by supporting full-text search against documents,

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate/>
@prefix rix: <http://omicspace.riken.jp/GRASQL/procedure/>
@prefix ris: <http://omicspace.riken.jp/GRASQL/statistics/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@let  %documentSet  rio:MEDLINE
@let  %researcher  rio:Researcher
@let  %keyword  "arabidopsis"
SELECT ?researcher ?k
WHERE {
  ?researcher  rip:hasDocument  ?doc ;
               rdf:type         %researcher .
  ?doc    EXT:rix:hasWord   %keyword  ;
          rip:publishYear   ?year ;
          rdf:type          %documentSet .
  FILTER (?year >= 2003)
}
EVALUATE ?k FOR ?researcher {
  ?k = ris:countDistinct(?doc)
}
ORDER BY DESC(?k)
```

Fig. 3. A GRASQL query that ranks researchers by k index using MEDLINE abstracts.

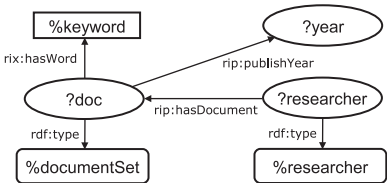

Fig. 4. RDF graph pattern specified in the WHERE clause in Figure 3.

LARQ (http://jena.sourceforge.net/ARQ/lucene-arq.html) is proposed. LARQ is an extension of ARQ that supports the specialized predicate `pf:textMatch` of RDF triples to access to the full-text search engine Apache Lucene (http://lucene.apache.org/). However, we would like a language that achieves all the features described above and allows writing statistical procedures explicitly with a query performing RDF graph pattern matching. Furthermore, in order to achieve both readability and descriptive capability, a language based on a standardized RDF query language and employing a programmable statistical computation description element is required. Thus, we have developed a novel language that satisfies these requirements. As our base language, we employ SPARQL because it is a specification of the W3C recommendation as of January 2008. Our language, termed GRASQL, contains the SELECT and CONSTRUCT statements of SPARQL extended by introducing EVALUATE clauses that denote a statistical computation method. Figure 5 shows an example of a GRASQL query with a SELECT statement.

The intuitive meaning of the EVALUATE clause `EVALUATE` p `FOR` $\bar{u}\{\bar{e}\}$ that appears just after the WHERE clause is as follows: In order to compute p for each set of solutions of $\bar{u}$ obtained by pattern matching in the WHERE clause, a set of equations $\bar{e}$ is evaluated over the aggregation a of solutions of all variables that appears in the WHERE clause. The values of p are added to the aggregation a of solutions. Other EVALUATE clauses are evaluated over the aggregation a of solutions obtained by all previous EVALUATE clause. In the example of Figure 5, `?r` means conditional probability $P(a|b)=a/b$ with two integer arguments a and b. The intuitive meaning of the query in the example is that mouse genes are ranked by each conditional probability $P_{?gene}(a_{?gene}|b_{?gene})$ of each mouse gene `?gene`, where $a_{?gene}$ is the number of MEDLINE documents that include the keyword 'diabetes' and associated with `?gene`, and $b_{?gene}$ is the number of MEDLINE documents associated with `?gene` using the statistical function

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate/>
@prefix rix: <http://omicspace.riken.jp/GRASQL/procedure/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@let  %keyword  "diabetes"
@let  %documentSet  rio:MEDLINE
@let  %geneSet  rio:MouseGene
SELECT ?gene ?r
WHERE {
  ?gene  rip:hasDocument  ?docIntersection ;
         rip:hasDocument  ?docGene ;
         rdf:type         %geneSet .
  ?docIntersection  EXT:rix:hasWord  %keyword ;
                    rdf:type         %documentSet .
  ?docGene  rdf:type  %documentSet .
}
EVALUATE ?r FOR ?gene {
  ?r = ?a/?b ;
  ?a = ris:countDistinct(?docIntersection) ;
  ?b = ris:countDistinct(?docGene)
}
ORDER BY DESC(?r)
```

Fig. 5. A sample GRASQL query that ranks mouse genes semantically associated with the keyword 'diabetes'. GRASQL extensions are highlighted in bold.

GRASQL supports a declaration of global constants for convenience of programming and readability. A constant is declared by an LET statement that appears just before a SELECT or a CONSTRUCT statement. The LET statement is written as `@let` `%`*constantName value*, where `%`*constantName* is the name of a constant that starts with '`%`', and *value* is its constant value.

In the SELECT statement, EVALUATE clauses follow the WHERE clause. The WHERE clause is a declaration of the condition for searching RDF graph patterns. In our example, the predicate `rip:hasDocument` in the WHERE clause takes an entity t as its subject and a document d as its object, which semantically means that the entity t is described in the document d. We suppose here that an RDF document containing the predicate `rip:hasDocument` is previously generated by applying a named entity recognition (NER) technique (Leser and Hakenberg, 2005) over mouse gene names and MEDLINE abstracts. We recompute the `rip:hasDocument` relationships so that they can be queried using simple RDF graph pattern matching. Another predicate `rix:hasWord` in the WHERE clause considers a document d as its subject and a string s as its object, which semantically means that the document d includes the string s.

A reserved word `EXT:` that appears just before a predicate such as `EXT:rix:hasWord` denotes that an external procedure associated with the predicate is called for matching RDF triples. The predicate `rix:hasWord` is used to call the full-text search engine Apache Lucene as with `pf:textMatch` in LARQ. Thus, the string s can be a query string for full-text search in Apache Lucene Query Language.

The EVALUATE clause is used for describing a statistical computation, written in the form `EVALUATE` p `FOR` $\bar{u}\{\bar{e}\}$, where p is a real or integer variable, $\bar{u}$ is a set of variables that appear in the WHERE clause and $\bar{e}$ is a set of equations that computes p. A set of equations $\bar{e}$ is denoted as equations connected with a semicolon '`;`', which can be described with arithmetic operators `+`, `-`, `*` and `/`, external statistical functions named by URIs, variables that appear in the WHERE clause, local variables in $\bar{e}$ and the variable p.

`ris:countDistinct`. An evaluation of the EVALUATE clause of the example is performed as follows:

(1) All values of triples (`?gene`, `?docIntersection`, `?docGene`) are first grouped into a set (`?docIntersection`, `?docGene`) of sub-solutions for each entity `?gene`.

(2) In order to compute `?r`, for each sub-solution, `?a=countDistinct(?docIntersection)` and `?b=count`

`Distinct(?docGene)` are evaluated so that the number of documents without duplication listed in `?docIntersection` and `?docGene` can be determined; then, `?a/?b` is evaluated with those computed values. The return value of the function is substituted for `?r`.

Then, by evaluating the ORDER BY clause, all sub-solutions of `?gene` are sorted by `?r` in descending order. Finally, the solution of the example is obtained as a set of mouse gene pairs and their concerned conditional probability values sorted by `?r` by evaluating the ORDER BY clause.

2.3 The statistical test used in the search

Our method discovers entities significantly related to a user's keyword using the documents associated with the entities. In this study, we call an entity or a document to clearly denote that the RDF name is a biomedical entity or a document, respectively. The simplest method for discovering entities is (1) full-text search over documents to find those containing the user's keyword, and then (2) obtaining entities associated with the documents found. This process is herein notated as *keyword → document → entity*. In order to compute the significance of association between each entity and a keyword, we have introduced a statistical test based on the number of shared documents. More concretely, for each entity, the search engine first generates a 2×2 contingency table consisting of the number of documents

(a) matching both the keyword and the entity
(b) matching the keyword but not matching the entity
(c) not matching the keyword but matching the entity and
(d) matching neither the keyword nor the entity.

Then, the engine applies a statistical test to the contingency table and computes a *P*-value or the significance of the test. Finally, all resultant entities are ranked by their *P*-values. We call the discovering method described above as single-association search, which is described as the query in Figure 6. The statistical function `ris:statisticalTest #FisherExactTest` computes the *P*-value by constructing a 2×2 contingency table $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ with its four arguments a, b, c, d and applies the Fisher's exact test to the contingency table.

A simple method of evaluation of the query shown in Figure 6 is a sequential evaluation of the WHERE, EVALUATE and ORDER BY clauses in this order. In this method, the WHERE clause is evaluated to obtain all RDF graphs satisfying the condition in the WHERE clause. Figure 7 shows the RDF graph pattern with all variables and constants appearing in the WHERE clause.

In practice, since the number of RDF graphs matching the pattern in Figure 7 may be huge, this simple method of evaluation requires the implementation of an optimization mechanism to achieve a functional language processor. Figure 8 is the chart that includes a Venn diagram of MEDLINE abstracts, which shows the relationship between each subset of MEDLINE abstracts and other RDF entities. This figure shows the primitive data structure for query searching as a set of relationships between each subset of MEDLINE abstracts and other entities such as `%keyword` and `?gene`, rather than the relationships between each MEDLINE abstract and other entities. In our example, in order to compute `?p`, only 4 subsets `?docAll`, `?docKey`, `?docIntersection` and `?docGene` of MEDLINE abstracts are necessary, which can be obtained by a specialized method of document search such as the full-text search technique. Since this approach does not require huge RDF graph space to compute statistical significance, it opens up a new possibility for realizing a practical language processing system of GRASQL. Furthermore, the results of statistical analysis can be stored as a named graph using the CONSTRUCT statement instead

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate/>
@prefix rix: <http://omicspace.riken.jp/GRASQL/procedure/>
@prefix ris: <http://omicspace.riken.jp/GRASQL/statistics/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@let  %keyword  "type 2 diabetes"
@let  %documentSet  rio:MEDLINE
@let  %geneSet  rio:MouseGene
SELECT ?gene ?p
WHERE {
  ?gene  rip:hasDocument  ?docGene ;
         rip:hasDocument  ?docIntersection ;
         rdf:type         %geneSet .
  ?docKey  EXT:rix:hasWord  %keyword ;
           rdf:type     %documentSet .
  ?docIntersection  EXT:rix:hasWord  %keyword ;
                    rdf:type     %documentSet .
  ?docGene  rdf:type  %documentSet .
  ?docAll  rdf:type  %documentSet .
}
EVALUATE ?p FOR ?gene {
  ?p = ris:statisticTest#FisherExactTest(?a,?b,?c,?d) ;
  ?a = ris:countDistinct(?docIntersection) ;
  ?b = ris:countDistinct(?docKey)-?a ;
  ?c = ris:countDistinct(?docGene)-?a ;
  ?d = ris:countDistinct(?docAll)-?a-?b-?c
}
ORDER BY ?p
```

Fig. 6. A GRASQL query for a single-association search using the Fisher's exact test as a method for computing statistical significance of the intersection `?docIntersection`.

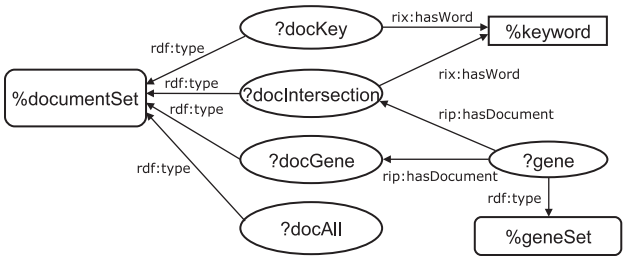

Fig. 7. RDF graph pattern satisfying the condition described in WHERE clause shown in Figures 6 and 9.

of the SELECT statement. Figure 9 shows a CONSTRUCT query that generates RDF graphs with blank nodes as shown in Figure 8. The generated named graph can be efficiently used in SPARQL as an input data as well as in GRASQL.

As another usage of statistical tests in a search, associations between entities and a keyword can be indirectly generated via entity–entity relationships associated with the documents. A typical example of entity–entity relationships is the co-citation frequencies of the entities in the documents. The significance of the association between two entities can be computed by a statistical test of the number of documents as well as a single-association search. That is, for each entity–entity relationship, a *P*-value is computed using a 2×2 contingency table that contains the number of documents

(a) matching both the entities
(b) matching the first entity but not matching the second entity
(c) not matching the first entity but matching the second entity and
(d) matching neither entities.

The entity–entity relationship can be obtained as a set of RDF triples using the query shown in Figure 10.

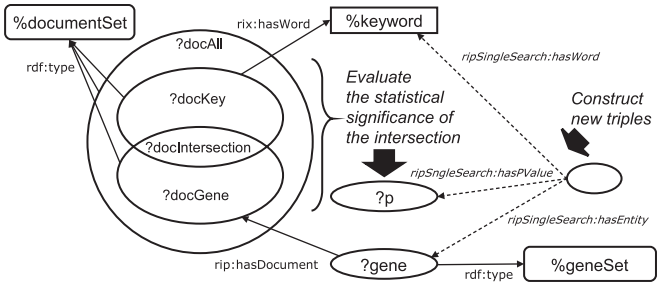

Fig. 8. A statistical diagram showing the relationship among the entities specified by the query in Figure 9. New RDF triples are constructed by the CONSTRUCT statement of the query.

We realize a double-association search for the connection *user's keyword* → *document* → *entity* → *document* → *entity* by applying entity–entity relationships to the resultant entities of a single-association search. The *P*-value P_d of the associated entity is computed by the following equation:

$$P_d = 1 - (1 - P_s)(1 - P_r) \tag{1}$$

where P_s is the *P*-value of the first single-association search, and P_r is the *P*-value of the second single-association search of the entity–entity relationship.

Furthermore, several search connections, *user's keyword* → *document* → $entity_1$ → *document* → $entity_2$, which reach the same entity $entity_2$ via a different entity $entity_1$ may be obtained. In this case, the *P*-value of the resultant entity $entity_2$ can be computed by the following equation:

$$P_{entity_2} = \prod_i P_{i,entity_2} \tag{2}$$

where $P_{i,entity_2}$, $(1 \le i \le n)$ are *P*-values of *n* connections that finally reach $entity_2$. This model is based on the idea that the solution containing several connections may play a more important role than the others. Another method is selecting the best connection by choosing the smallest *P*-value. In this case, the equation,

$$P_{entity_2} = \min_i(P_{i,entity_2}) \tag{3}$$

is applied for computing *P*-value. Examples of the multiple-association search are discussed in the next section.

3 RESULTS

3.1 Applications for typical search patterns in GRASQL in the biomedical field

In order to evaluate the descriptive capability of GRASE for practical biomedical searching with our approach, we write typical biomedical search patterns in GRASQL.

3.1.1 Double-association search using entity–entity relationships We start with a GRASQL query of double-association search described above. For the sake of convenience in enumerating examples, we first assume that a named graph http://omicspace.riken.jp/GRASQL/single/Mm/MEDLINE from a single-association search obtained by the query in Figure 9 is generated. Furthermore, we also assume that a named graph http://omicspace.riken.jp/GRASQL/relation/Mm/MEDLINE of entity–entity relationships obtained by the query in Figure 10 is generated.

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate/>
@prefix rix: <http://omicspace.riken.jp/GRASQL/procedure/>
@prefix ris: <http://omicspace.riken.jp/GRASQL/statistics/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix ripSigleSearch:
  <http://omicspace.riken.jp/GRASQL/singleSearch>
@let  %keyword  "type 2 diabetes"
@let  %documentSet  rio:MEDLINE
@let  %geneSet  rio:MouseGene
CONSTRUCT {
  []  ripSingleSearch:hasEntity  ?gene ;
      ripSingleSearch:hasWord    %keyword ;
      ripSingleSearch:hasPValue  ?p .
}
WHERE {
  ?gene  rip:hasDocument  ?docGene ;
         rip:hasDocument  ?docIntersection ;
         rdf:type         %geneSet .
  ?docKey  EXT:rix:hasWord  %keyword ;
           rdf:type     %documentSet .
  ?docIntersection  EXT:rix:hasWord  %keyword ;
                    rdf:type     %documentSet .
  ?docGene  rdf:type  %documentSet .
  ?docAll  rdf:type  %documentSet .
}
EVALUATE ?p FOR ?gene {
  ?p = ris:statisticalTest#FisherExactTest(?a,?b,?c,?d) ;
  ?a = ris:countDistinct(?docIntersection) ;
  ?b = ris:countDistinct(?docKey)-?a ;
  ?c = ris:countDistinct(?docGene)-?a ;
  ?d = ris:countDistinct(?docAll)-?a-?b-?c
}
```

Fig. 9. A GRASQL query including the CONSTRUCT statement. This CONSTRUCT statement is used to save the result of the statistical analysis described in the WHERE and EVALUATE clauses into a set of RDF graphs. In the CONSTRUCT statement, a blank node [] is used to describe the relationships among `?gene`, `%keyword` and `?p`, which is also used in SPARQL.

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate>/
@prefix ris: <http://omicspace.riken.jp/GRASQL/statistics/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix ripInference:
  <http://omicspace.riken.jp/GRASQL/inference>
@let  %documentSet  rio:MEDLINE
@let  %geneSet  rio:MouseGene
CONSTRUCT {
  []  ripInference:hasEntity1  ?gene1 ;
      ripInference:hasEntity2  ?gene2 ;
      ripInference:hasPValue   ?p .
}
WHERE {
  ?gene1  rip:hasDocument  ?docGene1 ;
          rip:hasDocument  ?docIntersection ;
          rdf:type         %geneSet .
  ?gene2  rip:hasDocument  ?docGene2 ;
          rip:hasDocument  ?docIntersection ;
          rdf:type         %geneSet .
  ?docGene1  rdf:type  %documentSet .
  ?docGene2  rdf:type  %documentSet .
  ?docIntersection  rdf:type  %documentSet .
  ?docAll  rdf:type  %documentSet .
}
EVALUATE ?p FOR ?gene1 ?gene2 {
  ?p = ris:statisticalTest#FisherExactTest(?a,?b,?c,?d) ;
  ?a = ris:countDistinct(?docIntersection) ;
  ?b = ris:countDistinct(?docGene1)-?a ;
  ?c = ris:countDistinct(?docGene2)-?a ;
  ?d = ris:countDistinct(?docAll)-?a-?b-?c
}
```

Fig. 10. A GRASQL query that builds RDF triples of co-citation relationships of mouse genes of MEDLINE abstracts.

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rix: <http://omicspace.riken.jp/GRASQL/procedure/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix owl: <http://www.w3.org/2002/07/owl#>
@prefix ripSingleSerarch:
  <http://omicspace.riken.jp/GRASQL/singleSearch/>
@prefix  ripInference:
  <http://omicspace.riken.jp/GRASQL/inference/>
@let  %keyword  "type 2 diabetes"
@let  %geneSet  rio:MouseGene
SELECT ?gene2 ?gene1 ?p
FROM NAMED
  <http://omicspace.riken.jp/GRASQL/single/Mm/MEDLINE>
FROM NAMED
  <http://omicspace.riken.jp/GRASQL/relation/Mm/MEDLINE>
WHERE {
  ?gene2  EXT:rix:inLocus  "Mm:*:*-*" .
  {
     ?x  ripInference:hasEntity2  ?gene2 ;
         ripInference:hasEntity1  ?gene1 ;
         ripInference:hasPValue   ?pInference .
  } UNION {
     ?gene2  owl:sameAs  ?gene1 .
     FILTER {?pInference = 0.0 }
  }
  ?y  ripSingleSearch:hasEntity    ?gene1 ;
      ripSingleSearch:hasKeyword  %keyword ;
      ripSingleSearch:hasPValue   ?pGene1 .
  ?gene1  rdf:type   %geneSet .
  ?gene2  rdf:type   %geneSet .
}
EVALUATE ?p FOR ?gene1 ?gene2 {
  ?p = 1-(1-?pInference)(1-?pGene1)
}
ORDER BY ?p
```

Fig. 11. A GRASQL query that discovers genes within a single interval. In this example, the genomic interval is specified for the whole mouse genome, that is, all mouse genes having positions are included in the gene search.

Using the named graphs http://omicspace.riken.jp/GRASQL/single/Mm/MEDLINE and http://omicspace.riken.jp/GRASQL/relation/Mm/MEDLINE, we write a query for the double-association search of connection *user's keyword* → *document* → *entity* → *document* → *entity*, shown in Supplement 2.

3.1.2 Selecting significant genes out of a group of genes In biological analyses, it is often necessary to select a few important genes out of a group of genes, e.g. those included in a gene cluster extracted by an analysis of a DNA microarray result (Hayashizaki 2003). When a set of mouse genes and a keyword are given, a query that ranks mouse genes in the given gene set against the keyword 'diabetes' is written as the query shown in Supplement 3.

3.1.3 Gene Ontology analysis When a set of mouse genes is given, we write a query that finds Gene Ontology (GO) terms (The Gene Ontology Consortium, 2006) ranked by *P*-value. This example is different from others because a statistical computation is performed over the number of entities instead of documents. In this example, we assume there is a document set `rip:MouseGeneDefinition` of gene definitions in which each definition includes at least a gene ID, and the relationship between GO terms and mouse genes are obtained as the predicate `rip:associatedWith`. The query in Supplement 4 is described for searching GO terms associated with mouse genes #1, #2 and #3.

3.1.4 In silico positional cloning *In silico* positional cloning is a technique of informatics used to identify genes by specifying phenotypic keywords such as disease names and locations on a chromosome (Toyoda and Wada 2004; Heida *et al*., 2004). The first example shown in Figure 11 is a monogenic case, i.e. discovering genes in a single genomic interval. In this example, the query includes the keyword 'type 2 diabetes' and the interval 'Mm:*:*-*', which denotes the whole mouse genome.

For convenience, the results of single- and double-association searches are combined by UNION into one data structure. The property `owl:sameAs` and a FILTER clause are used for an identity inference to combine the results of direct search into the results of double-association search.

The second example shown in Figure 12 is a digenic case, i.e. discovering genes in two genomic intervals. This example finds genes with an epistatic interaction in a tumour model mouse using the keyword 'cancer' and intervals (63 214 874, 111 011 533) of chromosome 9 and (25 275 696, 92 307 904) of chromosome 15 on the mouse genome (Cozma *et al*., 2002).

3.2 Experimental implementation

As discussed in several examples mentioned above, our language GRASQL has a description capability that is sufficient for practical biomedical problems. The next issue is the development of an effective system that can solve a problem written in GRASQL. We have experimentally implemented a prototype search engine called GRASE that performs single-, double- and triple-association searches. In order to develop the software component, we employ Apache Lucene, a rapid full-text search engine with a rich query language, for testing the predicate `rix:hasWord`.

3.2.1 Data preparation For the experimental evaluation of GRASE, we have partially extracted mouse gene–MEDLINE abstract relationships as RDF triples including the predicate `rip:hasDocument`. More concretely, we have first applied NER for gene names, gene symbols and synonyms of 20 000 MGI mouse genes (http://www.informatics.jax.org/) that have chromosomal positions against 16 million MEDLINE abstracts. Then, we have refined the results of NER by reading carefully extracted MEDLINE abstracts for each mouse gene by humans. The task of refinement is still in progress and has been partially finished (70%, 14 000 of 20 000 mouse genes) as of August 2007. Furthermore, using the refined results of NER, we have obtained 753 000 mouse gene–mouse gene relationships by evaluating the GRASQL query shown in Figure 10.

We have also generated researcher–MEDLINE abstract relationships over 5585 researchers in RIKEN as RDF triples including the predicate `rip:hasDocument`.

3.2.2 Distributed implementation Since a single-association search can be independently performed for each target entity in parallel, we utilize 20 distributed computers to realize a high-throughput search. Therefore, we have distributed data for each mouse gene and researcher, i.e. MEDLINE abstracts and mouse gene–mouse gene relationship data concerned with each distributed mouse gene and researcher are distributed among

```
@prefix rio: <http://omicspace.riken.jp/GRASQL/>
@prefix rip: <http://omicspace.riken.jp/GRASQL/predicate/>
@prefix rix: <http://omicspace.riken.jp/GRASQL/procedure/>
@prefix ris: <http://omicspace.riken.jp/GRASQL/statistics/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix ripSingleSerarch:
  <http://omicspace.riken.jp/GRASQL/singleSearch/>
@prefix  ripInference:

<http://omicspace.riken.jp/GRASQL/inference/>
@let  %keyword  "cancer"
@let  %documentSet  rio:MEDLINE
@let  %geneSet  rio:MouseGene
CONSTRUCT {
  []  ripSingleSearch:hasEntity   ?gene ;
      ripSingleSearch:hasKeyword  %keyword ;
      ripSingleSearch:hasPValue   ?p .
}
WHERE {
  ?gene  rip:hasDocument  ?docGene ;
         rip:hasDocument  ?docIntersection ;
         rdf:type         %geneSet .
  ?docKey  EXT:rix:hasWord  %keyword ;
           rdf:type     %documentSet .
  ?docIntersection  EXT:rix:hasWord  %keyword ;
                    rdf:type     %documentSet .
  ?docGene  rdf:type  %documentSet .
  ?docAll  rdf:type  %documentSet .
}
EVALUATE ?p FOR ?gene {
  ?p = ris:statisticalTest#FisherExactTest(?a,?b,?c,?d) ;
  ?a = ris:countDistinct(?docIntersection) ;
  ?b = ris:countDistinct(?docKey)-?a ;
  ?c = ris:countDistinct(?docGene)-?a ;
  ?d = ris:countDistinct(?docAll)-?a-?b-?c
}
;
SELECT ?gene1 ?gene2 ?p
FROM NAMED
  <http://omicspace.riken.jp/GRASQL/relation/Mm/MEDLINE>
WHERE {
  ?gene1  EXT:rix:inLocus  "Mm:9:63214874-111011533" .
  ?gene2  EXT:rix:inLocus  "Mm:15:25275696-92307904" .
  {
    ?x  ripInference:hasEntity2  ?gene1 ;
        ripInference:hasEntity1  ?gene2 ;
        ripInference:hasPValue   ?pInference .
  } UNION {
    ?x  ripInference:hasEntity2  ?gene2 ;
        ripInference:hasEntity1  ?gene1 ;
        ripInference:hasPValue   ?pInference .
  }
  ?y  ripSingleSearch:hasEntity   ?gene1 ;
      ripSingleSearch:hasPValue   ?pSingle1 ;
      ripSingleSearch:hasKeyword  %keyword .
  ?z  ripSingleSearch:hasEntity   ?gene2 ;
      ripSingleSearch:hasPValue   ?pSingle2 ;
      ripSingleSearch:hasKeyword  %keyword .
  ?gene1  rdf:type  %geneSet .
  ?gene2  rdf:type  %geneSet .
}
EVALUATE ?p FOR ?gene1 ?gene2 {
  ?p = 1-(1-?pInference)(1-ris:min(?pSingle1,?pSingle2))
}
ORDER BY ?p
```

Fig. 12. A GRASQL query that discovers genes of an epistatic interaction. This query evaluates the CONSTRUCT statement and then evaluates the SELECT statement with the result of the CONSTRUCT statement.

the computers so that parallel computing for single-, double- and triple-association searches is achieved.

3.2.3 Execution results The solutions to the problems shown in Figures 3, 11 and 12 obtained by GRASE are shown in Table 1.

By evaluating the query of Problem 1 shown in Figure 3 with the keyword 'arabidopsis', GRASE found 416 RIKEN researchers in 0.118 s. Kazuo Shinozaki, who was ranked first, is the Director of RIKEN Plant Science Center and the most cited researcher in plant and animal science (6055 citations in his 138 papers, http://www.in-cites.com/top/2007/first07-pla.html).

By evaluating the query of Problem 2 shown in Figure 11, GRASE found 4455 mouse genes in 2.227 s. *Adiponectin* ranked first by single-association search and *Leptin* ranked second by double-association search. Both of them are related to insulin resistance and have been inspired as a causable gene of type 2 diabetes.

In the solution for the query of Problem 3 shown in Figure 12, a gene pair *Cdc25a* and *Myc* was ranked first in 1.369 s. This result is also successful, because the molecular-level interaction previously reported as *Myc* was a factor involved in the transcription of *Cdc25a* (Cozma *et al*., 2002).

4 DISCUSSION AND CONCLUSION

In order to make use of not only well-formed knowledge in RDF but also non-well-formed publication data on the Semantic Web, we have introduced statistical notions into the existing RDF query language SPARQL using a literature mining technique for a vast number of documents written in a natural language. The core data structure in our method is that documents are linked with each entities accurately associated by NER with human refinements, namely curation tasks. The advantages of this simple structure are as follows:

- Facility of keyword selection; an arbitrary keyword appears in at least one document. Thus, a user can choose a keyword that is not necessarily related to an entity the user wants to find.
- Open-ended extensibility of documents; a new document can be added to the system if it is associated with at least one existing entity. Documents about an entity written from various viewpoints enrich the knowledge so that the entity can be linked to the user's keyword.
- Open-ended extensibility of entities; a new entity can be added if at least one document associated with it exists. Therefore, entities of different categories can be introduced, which allows association search among the entities.
- Open-ended extensibility of semantic knowledge; existing biomedical data in RDF format can be introduced directly into a GRASQL query.

Another advantage of our method is that it employs a *P*-value as a clear criterion that shows the significance between user's keyword and the resultant entity. Thus, it is also possible to introduce relations obtained by biological experiments instead of co-citation relations in a document set. Considering the advantages listed above, our methodology can be applied to data integration, which inferentially connects entities of different notations. One of our concrete applications is the realization of the Omic Space (Toyoda and Wada, 2004;

Table 1. The results for queries from Problems 1, 2 and 3 shown in Figures 3, 11 and 12, respectively

	Problem 1 (Figure 3)		Problem 2 (Figure 11)		Problem 3 (Figure 12)	
keyword	arabidopsis		type 2 diabetes		cancer	
genomic interval(s)	*none*		*all*		(1) Mm:9:63214874-111011533 (2) Mm:15:25275696-92307904	
execution time [sec]	0.118		2.227		1.369	
number of solutions rank	416		4455		26	
1	Kazuo Shinozaki	(117 papers)	*Adipoq*	($P = 4.35E-899$)	*Cdc25a* $\longleftrightarrow$ *Myc*	($P = 3.93E-87$)
2	Motoaki Seki	(71 papers)	*Adipoq* $\rightarrow$ *Lep*	($P = 4.35E-899$)	*Smad3* $\longleftrightarrow$ *EP300*	($P = 8.27E-70$)
3	Shigeo Yoshida	(51 papers)	*Adipoq* $\rightarrow$ *Retn*	($P = 4.35E-899$)	*Map2k1* $\longleftrightarrow$ *Ptk2*	($P = 5.80E-46$)
4	Kazuko Yamaguchi-Shinozaki	(36 papers)	*Pparg*	($P = 1.28E-803$)	*Rassf1* $\longleftrightarrow$ *Dap*	($P = 3.15E-39$)
5	Satoshi Tabata	(35 papers)	*Pparg* $\rightarrow$ *Ppara*	($P = 1.28E-803$)	*Smad3* $\longleftrightarrow$ *Foxh1*	($P = 2.13E-33$)

In the ranked list of Problem 2, the results of single-association search are shown as single entities, and the results of double-association search are shown as $gene_1 \longleftrightarrow gene_2$, where $gene_2$ is an inferred gene from the resultant gene $gene_1$ of the single-association search. In the rank list of Problem 3, the solution is shown as $gene_1 \longleftrightarrow gene_2$, where $gene_1$ is located in the first interval and $gene_2$ is located in the second interval. The results are obtained using 20 distributed computers that have an Intel Xeon 3.6 GHz CPU with 4 GB memory each.

Toyoda *et al*., 2007), which is a model of knowledge integration ranging from genomes to phenomes. By introducing various species of genes, metabolites and disease names as entities, a network among various omics entities can be obtained by inference. Furthermore, by adding documents that describe omics entities, the network can be extended with the expanded knowledge.

Regarding the implementation of our method, we have discussed GRASE, which substantially supports the logic of single- and multiple-association searches. In our evaluation of GRASE using mouse genes and MEDLINE abstracts, we have confirmed that GRASE performs single- and multiple-association searches in only a few seconds, although we specified a keyword that leads to thousands of solutions. Concerning the solutions of GRASE against the keywords of disease names, one of the causable genes was ranked first even though our curation task is still in progress (on August 2007) and partially finished (about 70%, 14 000 per 20 000 MGI mouse genes that have positions). These results confirm the possibility of producing a practical *in silico* positional cloning system implemented as a web service that integrates various omics entities and biomedical document sets. Our web service named PosMed (http://omicspace.riken.jp) is tentatively published using GRASE and the data described in this article.

Concerning the application of GRASE to *in silico* positional candidate gene search, our related works include existing methods and implementations of selecting disease gene candidates using literature-based discovery approaches. Comparisons among the existing methods such as BITOLA (Hristovski *et al*., 2005), Manjal (Sehgal and Srinivasan 2005) and LitLinker (Yetisgen-Yildiz and Pratt, 2006) have been discussed in (Weeber *et al*., 2005) and in (Ganiz *et al*., 2005). Although LitLinker computes a Z score to identify correlated terms, these systems do not support a statistical test for computing a score of discovered data. Thus, it can be said that our approach is novel in this community, which is based on P-values computed by the Fisher's exact test via tables of numbers of documents as correlation score between user's keyword and resultant genes for ranking.

Our future work will include the formal syntax specification, definition of the semantics of GRASQL and an implementation of the GRASE search engine that fully supports the functions extended by GRASQL as well as those of SPARQL. Furthermore, we aim to realize an *in silico* positional cloning system with fully curated association data between mouse genes and MEDLINE abstracts.

Conflict of Interest: none declared.

REFERENCES

Ashburner,M. *et al*. (2000) Gene ontology: tool for the unification of biology. *Nat. Gene.*, **25**, 25–29.

Berners-Lee,T. *et al*. (2001) The Semantic Web. *Sci. Am.*, **284**, 34–43.

Chong,E. *et al*. (2005) An Efficient SQL-based RDF Querying Scheme. In *Proceedings of the 31st International Conference on very large databases (VLDO'05)*. VLDB Endowment, Trondheim, Norway, pp. 1216–1227.

Cozma,D. *et al*. (2002) A bioinformatics-based strategy identifiers c-Myc and Cdc25A as candidates for the Apmt mammary tumor latency modifiers. *Genome Res.*, **12**, 969–975.

Ganiz,M. *et al*. (2005) Recent advances in literature based discovery. *Technical Report, Lehih University*, **LU-CSE-05-027**.

Hayashizaki,Y. (2003) RIKEN mouse genome encyclopedia. *Mechanisms of ageing and development*, **124**, 93–102.

Heida,N. *et al*. (2004) TraitMap: an XML-based genetic-map database combining multigenic loci and biomolecular networks. *Bioinformatics*, **20**, i152–i160.

Hirsch,J.E. (2005) An Index to Quantify an Individual's Scientific Research Output. In *Proceedings of the National Academy of Sciences of the United States of America*. National Academy of Science, Washington, USA. Vol. 102, pp. 16569–16572.

Hristovski,D. *et al*. (2005) Using literature-based discovery to identify disease candidate genes. *Int. J. Med. Inform.*, **74**, 289–298.

Jenssen,T.K. *et al*. (2001) A literature network of human genes for high-throughput analysis of gene expression. *Nat. Genet.*, **28**, 21–28.

Karvounarakis,G. *et al*. (2003) Querying the Semantic Web with RQL. *Computer networks*, **42**, 617–640.

Leser,U. and Hakenberg,J. (2005) What makes a gene name? Named entity recognition in the biomedical leterature. *Brief Bioinform.*, **6**, 357–369.

Manola,F. and Miller,E. (2004) RDF Primer. *World Wide Web Consortium*, Recommendation REC-rdf-primer-20040210. http://www.w3.org/TR/2004/REC-rdf-primer-20040210/ (Accessed on January 23, 2008).

Noruzi,A. (2005) Google Scholar: The New Generation of Citation Indexes. *Libli*, **55**, 170–180.

Page,L. *et al*. (1998) The pagerank citation ranking: bringing order to the web. *Stanford Digital Library Technologies Project*.

Prud'Hommeaux,E. and Seaborne,A. (2008) SPARQL Query Language for RDF, *World Wide Web Consortium*, Recommendation REC-rdf-sparql-query-20080115. http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/ (Accessed on January 23, 2008).

Seaborne,A. (2004) RDQL – A Query Language for RDF. *W3C Member Submission 9 January 2004*, http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/.

Seaborne,A. (2006) ARQ – A SPARQL Processor for Jena, http://jena.sourcefourge.net/ARQ/ (Accessed on January 23, 2008).

Sehgal,A.K. and Srinivasan,P. (2005) Manjal - A Text Mining System for MEDLINE. In *Proceedings of the 28th Annual International ACM SIGIR*. ACM, Salvador, Brazil, p. 680.

The Gene Ontology Consortium (2006.) The Gene Ontology (GO) project in 2006. *Nucleic Acids Res.*, **34**, D322–D326.

Toyoda,T. and Wada,A. (2004) Omic space: coordinate-based integration and analysis of genome ~ phenomic interactions. *Bioinformatics*, **20**, 1759–1765.

Toyoda,T. *et al*. (2007) OmicBrowse: a browser of multidimensional omics annotations. *Bioinformatics*, **23**, 524–526.

Weeber,M. *et al*. (2005) Online tools to support literature-based discovery in the life sciences. *Brief Bioinform.*, **6**, 277–286.

Yetisgen-Yildiz,M. and Pratt,W. (2006) Using statistical and knowledge-based approaches for literature. based discovery. *J. Biomed. Inform.*, **39**, 600–611.