The Aristotle Semantic Network Technology

Ernst Kretschmann
European Bioinformatics Institute
Cambridge, CB10 1SD, United Kingdom

and

Astrid Rakow
European Bioinformatics Institute
Cambridge, CB10 1SD, United Kingdom

and

André Hackmann
European Bioinformatics Institute
Cambridge, CB10 1SD, United Kingdom

and

Rolf Apweiler
European Bioinformatics Institute
Cambridge, CB10 1SD, United Kingdom

Abstract

A novel way to represent knowledge using extended semantic networks as underlying model is presented. This approach extends the semantic network idea from its frequent application in presenting human-readable data correlations into a basis for a general computerized data modelling technology. The method was successfully applied using biological data from the Swiss-Prot and TrEMBL sections of the UniProt knowledge base to facilitate high throughput data mining applications. It is also shown that the method can be effectively used to automatically generate the relational implementation of a persistent warehousing system containing data from heterogeneous resources. The approach is extended to an object-oriented level, including the automated creation of object-relational mappings between objects held as data structures in an application framework and the underlying relational data warehouse component. The technology is used as persistency level of parts of the UniProt web site (see **http://www.ebi.uniprot.org**).

Keywords: Knowledge Representation, Data Modelling, Semantic Networks, Data Warehousing, Applications in Biology.

1. Introduction

We present a data modelling approach using semantic networks to represent knowledge, which proved to be capable of storing and retrieveing data of high complexity in an effective and efficient manner. It can also be automatically translated into a relational schema, facilitating a smooth object-relational translation that keeps maintenance on a single level. Persistency for read purposes can be generated without human interaction, enhancing quick implementations of data warehousing systems. The technology was successfully implemented to model highly complex data in the Swiss-Prot and TrEMBL databases, which are currently being unified with the PIR [1] database under the UniProt [2] initiative into an integrated protein knowledgebase. Rigid object-oriented or relational data models of this data required high maintenance in the past to keep up to date with the progressing semantics of protein annotation.

Additionally, a multitude of specialized protein databases exist, for which UniProt will be a central reference point to which they link, and from where many of them are cross-referenced in return. To be able to make maximum use of the UniProt effort in terms of data retrieval and data mining, a queryable data warehouse system is required that provides quick, easy, and reliable access to the contents of the knowledgebase. The possibility to seamlessly integrate information from external resources alongside knowledgebase data is essential. To facilitate this, a data model was needed that reflected the fine grained and complex structures of protein data, a task that proved to be non trivial to achieve in object-oriented and relational ways. With the presented technology based on semantic networks, a protein data model could be implemented that resolved the following inherent difficulties in representing protein data. To model data sources of other

origins will in general include similar problems that might be resolved by the use of the presented technology.

Conflicting and supportive data

Cross-referencing between data sources is in many cases not maintained in a strictly synchronized way. For example, Swiss-Prot and the InterPro [3] database of protein families, domains and functional sites cross-reference each other. According to the InterPro content of IPR000005, Swiss-Prot protein P40931 (*Myeloproliferative leukemia protein*) contains a *Helix-turn-helix, AraC type* domain. This domain hit was considered as false positive in Swiss-Prot, which lead to the removal of the cross-reference to IPR000005 from the Swiss-Prot entry, while the reference from InterPro to Swiss-Prot still exists. A query that selects all proteins from this particular family using a data warehouse containing both resources will return ambiguous results depending on the direction the cross-reference is represented.

Some data is supportive rather than conflicting. In the case of entry Q44501, the protein name is *"Dimeric (2Fe-2S) protein"* in TrEMBL, and *"ferredoxin [2Fe-2S] fesII"* in the PIR. For some mining purposes both names need to be taken into account, while for statistical analysis of the individual databases only the corresponding name is required. In a single data warehouse, supportive data needs to be tagged with a pointer to its origin, so that queries in both integrative and individual ways are possible.

Fine grained data structures and implicit semantics

Sequence data is firmly rooted in the flat file tradition. The Swiss-Prot flat file has evolved over more than 15 years into a complex structure, containing detailed information in a human readable textual format.

```
...
OS   Petunia hybrida (Petunia),
OS   Daucus carota (Carrot),
OS   Lilium longiflorum (Trumpet lily), and
OS   Bryonia dioica (Red bryony).
...
...  (References 1-4 stripped out)
...
RN   [5]
RP   SEQUENCE FROM N.A.
RC   SPECIES=B.dioica;
RX   MEDLINE=94072731; PubMed=8251636;
RA   Galaud J.-P., Lareyre J.-J., Boyer N.;
RT   "Isolation, sequencing and analysis of the expression
RT   of Bryonia calmodulin after mechanical perturbation.";
RL   Plant Mol. Biol. 23:839-846(1993).
...
FT   CONFLICT    136    136        I -> T (IN REF. 5).
...
SQ   SEQUENCE   148 AA;  16716 MW;  C62B515E3AC8833C CRC64;
     ADQLTDDQIS EFKEAFSLFD KDGDGCITTK ELGTVMRSLG QNPTEAELQD
     MINEVDADGN GTIDFPEFLN LMARKMKDTD SEEELKEAFR VFDKDQNGFI
     SAAELRHVMT NLGEKLTDEE VDEMIREADV DGDGQINYEE FVKVMMAK
//
```

Fig. 1: Interconnected data in a Swiss-Prot entry. Citations are presented in reference blocks (grey background) and linked to various other data items like organisms (OS) and sequence features (FT).

As biological research advanced, the original syntax of the flat file was not able to represent all the information units scientists liked to find in this database. To represent detailed annotations, textual structures were gradually introduced, which pointed from almost any arbitrary data item to any other within the entry. A quite remarkable example of that complexity is protein entry P27162 (*Calmodulin 1*), of which some parts of the flat file entry are shown in Fig. 1.

The protein described in this entry can be found in the four plants mentioned in the OS lines of the flat file entry. There are five literature reference blocks in this entry, each containing an individual citation. The organism, on which the citation reports is annotated in the RC line of a reference. The citation in the example refers to only one of the plants, namely *B.dioica*. The authors report a modification in the protein sequence of this plant, in which they found a Threonine (T) instead of Isoleucine (I) at position 136 in the protein sequence. This fact is found in the FT line. The information extracted from *Galaud et al.* is therefore distributed over several data items, most of which is found in the reference block, some in the OS line, and some in the feature table. A data model of this protein entry needs to store the links between those data items. That way, for instance, the link number to citation five in the FT line can be increased automatically, if a new literature reference is inserted before the citation it refers to.

The internal version of the TrEMBL flat file contain additional information that facilitates tracking annotation units to its sources, a feature needed for bulk clean-ups in the work flow. This sort of information is kept in evidences linked by evidence tags:

```
...
KW   NAD{EA1,EI3}, Oxidoreductase{EA1}; Plastoquinone{EA2}.
...
**EV EA1; RuleBase; -; RU000317V2.40; 18-NOV-2002.
**EV EA2; RuleBase; -; RU000319V1.79; 18-NOV-2002.
**EV EI3; EMBL; -; CAB99441.1; 07-FEB-2002.
...
//
```

Fig. 2: Internal section and evidence tags in an entry from TrEMBL.

The Keywords *"Oxidoreductase"* and *"Plastoquinone"* were inserted by the RuleBase automated annotation system, from annotation rules RU000317V2.40 and RU000319V1.79, respectively. The tags EA1 and EA2 establish the link between the keyword annotations and the resource that produced them. The *"NAD"* keyword contains multiple evidences: it was imported from the EMBL data bank, which is described by the EI3 evidence line, and is additionally confirmed by RuleBase RU000317V2.40, indicated by EA1.

Frequent format changes

Over the last decade exponential growth of protein data has been observed, doubling the number of entries in the Swiss-Prot and TrEMBL databases about every 18 months. Not only have protein numbers increased, but so has the knowledge about protein properties. Swiss-Prot, TrEMBL, and their successor UniProt are committed to presenting this knowledge to users, resulting in frequent format extensions and semantic changes. This produces a large maintenance overhead, of which mainly the persistency level, i.e. the relational model of the data, is affected.

Request for supplements

For some data warehousing applications, the content of an entry in UniProt needs to be enhanced with data from additional sources. For mining applications, databases such as CluSTr [4] are highly valuable. This resource is not cross-referenced in UniProt, but CluSTr refers to UniProt and can hence be linked to the corresponding entries. The presented data structure enables a quick and easy inclusion of such additional data sources.

2. Semantic Networks with Annotated Nodes and Edges

The basic idea of representing complex relations inside data sets in semantic network representation goes back to the sixties [5]. Applications of this are mind maps [6] and concept maps [7], which are frequently used for brainstorming purposes in a graphical way rather than as data models. Mind maps use textual annotation on the nodes, and concept maps use textual annotation on both nodes and edges of the graph. Those structures cannot represent data in a computerized way, since a human intervention is needed to extract and understand the inherent meaning of the diagrams that are represented in a mere textual way. To allow structural analysis of the graph, semantic networks were enriched by additional expressive means.

Typisation of Nodes and Edges

The concepts of object orientation such as inheritance or polymorphism can be used to structure the implementing data types. General types, such as the AristotleNodes and AristotleEdges themselves can be used to describe general properties, while the extending classes encapsulate specialized data items.
The methods getNodes(Class nodeType) and getEdges(Class edgeType) of the Aristotle class (see Fig. 3) will only return those nodes and edges of the given type of the parameter Class. This enhances simple queries to select all nodes or edges of general or special types.

Empty Nodes and Annotated Nodes

Empty nodes are introduced to keep the real world entities and their references apart, giving the structure of the data an additional dimension. Objects in the real world interact with each other, a fact that often has to be reflected in data structures. For example, objects do have names so they can be identified, but these are mere reference structures that do not affect the inherent properties of the objects themselves or their interactions. References have textual or numerical annotations like the character string of a name. All that is known about a real world entity itself can be regarded as references to the objects rather then a property. This at first sight artificial construct can be used to handle entities where there are mutually supportive references (see Fig. 5).

Multiple Edges between Nodes

Real world entities often refer to each other in more than one way, which frequently needs to be represented in the object model of those entities. The Aristotle model uses multiple edges between nodes to refer between the same two entities to be able to represent such cases.

3. Aristotle Semantic Networks

An Aristotle semantic network is based on a graph structure consisting of two kinds of nodes and a single kind of edge (see Fig. 3).

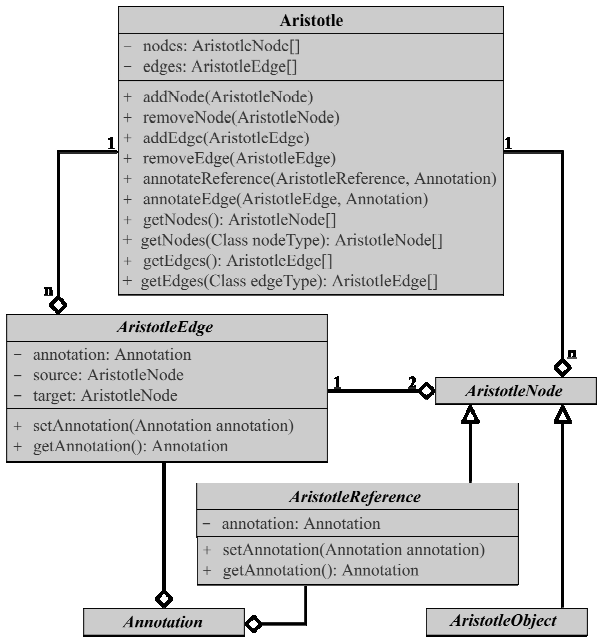

Fig. 3: The UML model of an Aristotle semantic network. A network consists of an arbitrary number of nodes and edges. Reference nodes and edges can be annotated, while objects remain empty. All internal objects are abstract and can be typed using inheritance.

Edges and nodes of AristotleReferences can contain annotation, while AristotleObjects contain no further structure at all. AristotleObjects and their descendants are empty structures that can be used as hubs to which information in the form of AristotleReferences are linked using AristotleEdges. All entities except the semantic network itself are abstract and need to be extended by structures representing application data.

4. Extensional Components

Controller Unit

The network itself is kept general to be able to model a variety of nodes or edges as long as they are extensions of AristotleEdge and AristotleNode. This set up does not prevent the introduction of links and annotations in an uncontrolled and unstructured fashion. To avoid this, the traditional separation between data model and controller unit is a valuable extension of the Aristotle idea. It takes care of a clean mapping between real world data items and their representations in the network.

Automatic Translation to Relational Database Schema

Semantic networks model relations between entities are, by nature, similar to relational schemas. Edges and nodes can be mapped to a database implementation automatically, facilitating the storage of the complete content of the network structure in a relational way.

AristotleNodes can be mapped to tables and AristotleEdges to foreign-key relationships between tables in a traditional data-warehouse snowflake/star schema. In order to entirely map semantic network relations to tables, the introduction of typed relations extended the classical approach. Therefore, a three-way relation between two AristotleNode tables and one AristotleEdge table is used to represent a typed relationship between two nodes. For most models, basic annotation implementations containing simple data items like character strings, numbers or dates are sufficient. An automated translation of those annotations can easily be achieved without the usage of further object-relational mapping software.

5. Results

Using the semantic model, the difficulties described in the introduction can easily be resolved. As with other types of modelling there is no unique way to represent data in an Aristotle semantic network. The following diagrams contain crude visualisations of the solutions. The focus is on applications operating on the data rather than humans reading it. The potential of the approach in terms of being able to query the structure are discussed.

Fine grained vs. general queries

It is essential to be able to query all entities of a general data type in the vast network representation of a protein entry. For instance, the set of all features of the protein sequence are needed to understand the topology of the protein. Individual querying is also required, for example to be able to query all sequence conflicts reported in the literature.

To achieve this, the inheritance mechanism on nodes was used. The abstract SequenceFeature class, which extends an AristotleNode was introduced to generally represent such features. It is modelled with a start and end position and a textual description and always refers to the protein sequence. The individual feature types present in UniProt extend this abstract class. Most of theses features do not go beyond the definition of the SequenceFeature itself, but some add data fields of their own to represent additional pieces of information.

The UniProt model also takes advantage of edge inheritances, which are used to describe the nature of the link between data items. Some annotations are added due to experimental evidences, a fact that can be expressed by letting all those edges extend an abstract ExperimentalEvidence edge. Others are merely predicted, which can be expressed by extending an abstract PredictedAnnotation edge. Realizations of these edges contain the exact nature of the experiment or predictor that was used to generate the particular annotation and can be queried individually.

Conflicting data

The case of out of sync cross-references can be modelled using the presence and absence of edges. All edges in the network are directed, thus two equal references between the same nodes pointing in opposite directions are used to express the fact that references are in synch. Out of synch data is represented by a single edge.

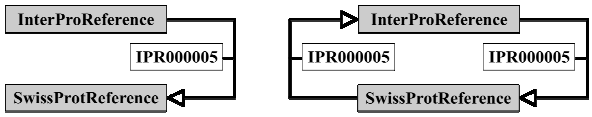

Fig. 4: The modelling of data conflicts. Left: The protein contains the domain only according to the content of InterPro. Right: The protein contains the domain according to both, Swiss-Prot and InterPro.

This means queries asking for all proteins contained in InterPro and belonging to IPR000005 can be kept distinct from queries asking for all proteins from Swiss-Prot cross-referencing InterPro IPR000005 (see Fig. 4).

Supportive data

The concept of empty nodes representing real world entities can be used to collect data from various sources referring to a single annotation type.

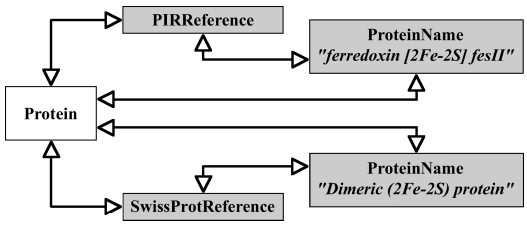

Fig. 5: Modelling supportive data. A single protein AristotleObject (white) is referred by two distinct but valid names from two separate resources.

This structure can be queried to retrieve not only names known for a protein, but also which particular name is available in which of the individual resources (see Fig 5).

Complex structures

The complexity of the data from the flat file example in the introduction (see Fig. 1) can be translated into a semantic network of which parts are shown in Fig. 6.

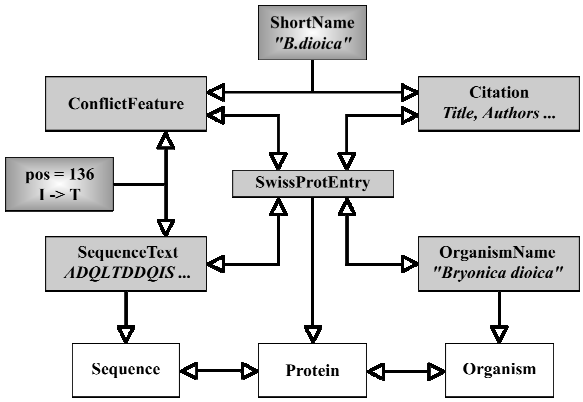

Fig. 6: Complex data dependencies between real world entities (white), references (grey) and annotations on nodes and edges.

The example shows the relationship between objects, in this case the protein, the organisms where it can be found, and its sequence. Some references to those objects like one organism name and a citation are also presented. Object–object relations, object-references relations and reference–reference relations are typed (not shown) and can be annotated with additional information.

Model and Controller

Controller classes were introduced that ensured the network semantically represented the real world protein data in a well-structured way. The following code fragment (see Fig. 7) shows the use of a controller of an Aristotle semantic network implemented in the Java language. All the methods in the entry object are mapped to operations, i.e. queries and modifications of the underlying graph structure.

```
// the controller
Entry entry = ...;

// the model
Aristotle a = ...;

// controller - model communication established
entry.setAristotle(a);

// the taxonomy of the organism from where the
// protein was extracted this data item is
// annotated in all UniProt entries
Vector taxonomy = entry.getTaxonomy();

// condition
if( entry.hasInterProHit("IPR000001") &&
    taxonomy.contains("Viridiplantae")) {

  // implication
  e.addKeyword("Chloroplast");
}
```

Fig. 7: Code fragment showing the use of the model – controller separation. An `Entry` *object is the controller class to an Aristotle semantic network, which contains all the information about a particular protein.*

Minimisation of Maintenance

The approach proved to reduce maintenance work considerably. The example below shows the format of the recently introduced Swiss-Prot alternative splicing annotation. This was the first time that the annotation items Comment (CC) and Feature (FT) were linked by an annotation structure.

```
...
CC   ALTERNATIVE PRODUCTS:
CC   Event=Alternative splicing; Named isoforms=3
CC   Name=A2;
CC     IsoId=P34007-1; Sequence=Diplayed;
CC   Name=A1;
CC     IsoId=P34007-2; Sequence=VSP_004606; VSP_004607
CC   Name=A3;
CC     IsoId=P34007-3; Sequence=VSP_004608;
...
FT   VARSPLIC  196   198   GRR -> DVR (In isoform A1).
FT                         /FTId=VSP_004606
FT   VARSPLIC  197   213   RRRESGKKRKRKRLPT -> TLLPAPG
FT                         GVHPQGCLRAHDGCQSSRNHMQALGWK
FT                         KKN (In isoform A3).
FT                         /FTId=VSP_004608
FT   VARSPLIC  199   213   Missing (In isoform A1).
FT                         /FTId=VSP_004607
...
//
```

Fig. 8: Illustration of the introduction of the Swiss-Prot ALTERNATIVE PRODUCTS comment. In general it links to one or more VARSPLIC Features. In this example isoform A1 is obtained through two sequence modifications, while A3 requires only one.

Using the Aristotle technology, the link between the Swiss-Prot *"ALTERNATIVE PRODUCTS"* comment in the CC line and the *"VARSPLIC"* Feature in the FT line required the introduction of a new Swiss-Prot AlternativeProducts comment extending an AristotleReference and a new AristotleEdge between those Comments and the respective Feature. No modifications had to be introduced in the relational schema, since no former parts of the model had to be changed and the new information entities were automatically added to the schema using a direct model to relational schema translation.

6. Discussion

The Aristotle technology was successfully introduced in data mining applications on UniProt, where each entry was represented in a semantic network that could be accessed and modified by object-oriented access classes. A standard data-mining algorithm was used to learn from well-trusted information contained in Swiss-Prot entries and to produce decision trees describing these data [8]. Those decision trees are then applied on entries in TrEMBL. All the necessary data items were stored as Aristotle semantic networks both on the application and persistency level.

Using this approach it was possible to apply about 30.000 annotation rules from the Spearmint project on more than 1.000.000 database entries in less than 2 hours. This reduced application time by an order of two magnitudes compared to the application of 450 rules, as described in Fleischmann et al. [9]. The results of the annotation runs are stored together with the data originally present in the entries inside an Aristotle data warehouse. Annotation stemming from human curators and automatically gener-

ated annotation is kept apart using different types of AristotleEdges to link between the entry and the respective annotation items. That way statistical analyses on automated annotation and data from manual curation can be generated separately and browsed individually. Since the described mechanism to automatically generate the persistency layer was used, changes and additions to the flat file formats could be accommodated with little maintenance.
The technology to automatically translate Aristotle semantic networks into relational representation is used as the read only back end of the UniProt data base. Data in form of Aristotle objects representing protein entries can be browsed at **http://www.ebi.uniprot.org**.
The use of Aristotle semantic networks to model, modify and store data can be effective in a computationally expensive and data intensive domain such as protein biology. Using the presented technologies, implementations of object relational mappings on complex data items proved superior to traditional approaches. Maintenance could be focused to a single layer, namely the correlation between the semantic network itself and the object-oriented wrapper classes around it.

7. Conclusion

We suggest consideration of this technology in any problem field that

1. deals with data that undergoes frequent changes.
2. requires a read only persistency level and object-relational mappings.
3. needs to assemble and link data from heterogeneous sources.

The approach fits well into traditional model-view-controller designs. Aristotle models can be easily transferred to object-oriented and relational structures, hence avoiding the necessity for additional object relational-mappings.
As for the controller interface to the data model a query interface to the physical relational database management system will be required in the future. If the data model is modified, the relational schema is bound to change accordingly, rendering legacy SQL queries worthless. This maintenance overhead can be avoided by implementing access classes or an application tailored query engine to the warehouse that serves external requests. Remote components that make use of such a query engine need not be maintained in case of data format changes as long as the server is kept up to date with the modifications in the schema.

8. Acknowledgements

This work was supported by the National Institutes of Health (NIH) grant 1 U01 HG02712-01.

9. References

[1] C.H. Wu, L.L. Yeh, H. Huang, L. Arminski, J. Castro-Alvear, Y. Chen, Z.Z. Hu, R.S. Ledley, P.K. Baris, E. Suzek, C. R. Vinayaka, J. Zhang, W.C. Barker, "The Protein Information Resource", **Nucleic Acids Research**, Vol. 31, pp. 345-347, 2003.

[2] R. Apweiler, C. O'Donovan, M.J. Martin, "Protein Sequence Database Resources", **Screening Trends in Drug Discovery**, Vol. 4, pp. 33-35, 2003.

[3] N.J. Mulder, R. Apweiler, T.K. Attwood, A. Bairoch, D. Barrell, A. Bateman, D. Binns, M. Biswas, P. Bradley, P. Bork, P. Bucher, R.R. Copley, E. Courcelle, U. Das, R. Durbin, L. Falquet, W. Fleischmann, S. Griffiths-Jones, D. Haft, N. Harte, N. Hulo, D. Kahn, A. Kanapin, M. Krestyaninova, R. Lopez, I. Letunic, D. Lonsdale, V. Silventoinen, S.E. Orchard, M. Pagni, D. Peyruc, C.P. Ponting, J.D. Selengut, F. Servant, C.J.A. Sigrist, R. Vaughan, E.M. Zdobnov, "The InterPro Database, 2003 brings increased coverage and new features", **Nucleic Acids Research**, Vol. 31, pp. 315-318, 2003.

[4] E.V. Kriventseva, W. Fleischmann, E.M. Zdobnov, R. Apweiler, "CluSTr: a database of clusters of Swiss-Prot+TrEMBL proteins". **Nucleic Acids Research**, Vol. 29, No. 1, pp. 33-36, 2001.

[5] M. Quillian, **Semantic Memory**. In M. Minsky, editor, Semantic Information Processing, pp. 227-270, MIT Press, Cambridge, MA, 1968.

[6] T. Buzan, **The Mind Map Book**, 1996.

[7] J.D. Novak, Concept maps and Vee diagrams: Two metacognitive tools for science and mathematics education, **Instructional Science**, Vol. 19, pp. 29-52, 1990

[8] E. Kretschmann, W. Fleischmann, R. Apweiler, "Automatic rule generation for protein annotation with the C4.5 data mining algorithm applied on Swiss-Prot", **Bioinformatics**, Vol. 17, pp. 920-926, 2001.

[9] W. Fleischmann, S. Moeller, A. Gateau, R. Apweiler, "A novel method for automatic and reliable functional annotation", **Bioinformatics**, Vol. 15, pp. 228-233, 1999.