Flexible Integration of Molecular-biological Annotation Data: The GenMapper Approach

Hong-Hai Do[1], Erhard Rahm[2]
[1] Interdisciplinary Centre for Bioinformatics, [2] Department of Computer Science
University of Leipzig, Germany
www.izbi.de, dbs.uni-leipzig.de
{hong, rahm}@informatik.uni-leipzig.de

Abstract. Molecular-biological annotation data is continuously being collected, curated and made accessible in numerous public data sources. Integration of this data is a major challenge in bioinformatics. We present the GenMapper system that physically integrates heterogeneous annotation data in a flexible way and supports large-scale analysis on the integrated data. It uses a generic data model to uniformly represent different kinds of annotations originating from different data sources. Existing associations between objects, which represent valuable biological knowledge, are explicitly utilized to drive data integration and combine annotation knowledge from different sources. To serve specific analysis needs, powerful operators are provided to derive tailored annotation views from the generic data representation. GenMapper is operational and has been successfully used for large-scale functional profiling of genes. Interactive access is provided under http://www.izbi.de.

1. Introduction

Over the past few years, genomes of several organisms, especially the human genome, have been completely sequenced. Now the focus of genomic research has shifted to understand how genes and ultimately entire genomes are functioning. The knowledge about molecular-biological objects, such as, genes, proteins, intra- and inter-cellular pathways, etc., is typically encoded by a large variety of data commonly called annotations. Such annotations are continuously collected, curated, and made available in numerous public data sources. A current survey lists more than 500 such databases [14]. Furthermore, an increasing number of ontologies is maintained, mostly in the form of standardized vocabularies and hierarchical taxonomies. Typically, objects in one source are annotated by information in other sources and ontologies in the form of cross-references (web-links) [8, 7, 11]. A few sources focus on sequence-based objects and uniformly map them onto the genome of a particular species for the visual comparison and correlation of co-located objects [2, 6, 17].

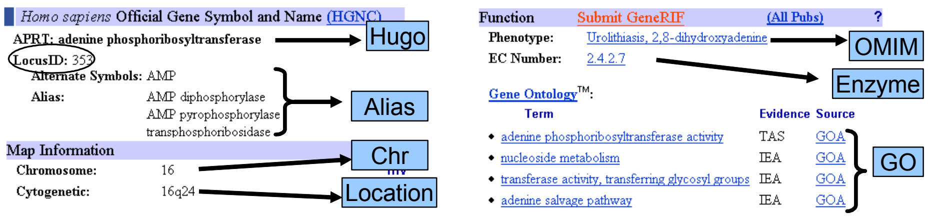
Many applications such as functional gene profiling, gene expression analysis, protein analysis, etc., require molecular-biological objects and their annotations to be integrated from different sources and made accessible for queries and data mining. This integration task is a major problem since annotation data is highly diverse and only structured to some extent. Moreover, the number and contents of relevant sources are continuously expanding [26]. The use of web-links or the display of related objects on the genome represent first integration approaches, which are very useful for interactive navigation. However, they do not support automated large-scale analysis tasks. While more advanced integration approaches are needed, it is important to preserve and utilize the semantic knowledge about relationships between objects, which are typically established by domain experts (curators).

A survey of representative data integration systems in bioinformatics is given in [26]. Current solutions mostly follow a data warehouse (e.g. IGD [30], GIMS [29], DataFoundry [16]) or federation approach (e.g. TAMBIS [21], P/FDM [24]) with a physical or virtual integration of data sources, respectively. These systems are typically built on the notion of an application-specific global schema to consistently represent and access integrated data. However, construction and maintenance of the global schema (schema integration, schema evolution) are highly difficult and do not scale well to many sources. DiscoveryLink [22] and Kleisli [31] also follow the federation approach but their schema is simply the union of the local schemas, which have to be transformed to a uniform format, such as relational (DiscoveryLink), or nested relational (Kleisli). A general limitation of these systems is that existing cross-references between sources are not exploited for semantic integration.

SRS [19] and DBGET/LinkDB [20] do not follow a global schema approach. In these systems, each source is replicated locally as is, parsed and indexed, resulting in a set of queryable attributes for the corresponding source. While a uniform query interface is provided to access the imported sources, join queries over multiple sources are not possible. Cross-references can be utilized for interactive navigation, but not for the generation and analysis of annotation profiles of objects of interest. Recently, Kementsietsidis et al [23] established a formal representation of instance-level mappings, which can be obtained from the cross-references between different sources, and proposed an algorithm to infer new mappings from existing ones.

GenMapper (*Gen*etic *Mapper*) represents a new approach to flexibly integrate a large variety of annotation data for large-scale analysis that preserves and utilizes the semantic knowledge represented in cross-references. The key aspects of our approach are the following:

- GenMapper physically integrates all data in a central database to support flexible, high performance analysis across data from many sources.
- In contrast to previous data warehouse approaches, we do not employ an application-specific global database schema (e.g. a star or snowflake schema). Instead, we use a generic data model called *GAM (Generic Annotation Management)* to uniformly represent object and annotation data from different data sources, including ontologies. The generic data model makes it much easier to integrate new data sources and perform corresponding data transformations, thereby improving scalability to a large number of sources. Moreover, it is robust against changes in the external sources thereby supporting easy maintenance.
- We store existing cross-references between sources (mappings) and associations between objects and annotations, and exploit them to combine annotation knowledge from different sources.
- To support specific analysis needs and queries, we derive tailored *annotation views* from the generic data representation. This task is supported by a new approach utilizing a set of high-level operators, e.g. to combine mappings. Results of such operators that are of general interest, e.g. new mappings derived from existing mappings, can be materialized in the central database. The separation of the generic data representation and the provision of application-specific views permits


Homo sapiens Official Gene Symbol and Name (HGNC)

APRT: adenine phosphoribosyltransferase

LocusID: 353

Alternate Symbols:	AMP
Alias:	AMP diphosphorylase AMP pyrophosphorylase transphosphoribosidase

Map Information

Chromosome:	16
Cytogenetic:	16q24

Function Submit GeneRIF (All Pubs) ?

Phenotype:	Urolithiasis, 2,8-dihydroxyadenine
EC Number:	2.4.2.7

Gene Ontology^{TM}:

Term	Evidence	Source
• adenine phosphoribosyltransferase activity	TAS	GOA
• nucleoside metabolism	IEA	GOA
• transferase activity, transferring glycosyl groups	IEA	GOA
• adenine salvage pathway	IEA	GOA

Figure 1. Sample annotations from LocusLink

GenMapper and its (imported and derived) data to be used for a large variety of applications.

GenMapper is fully operational and currently integrates more than 60 public sources, including those for gene annotations, such as LocusLink [8] and Unigene [12], and for protein annotations, such as InterPro [7] and SwissProt [11]. Furthermore, it includes various sub-divisions of NetAffx, a vendor-based data source of annotations for genes used in microarray experiments [1]. GenMapper has been successfully used for large-scale functional profiling of genes [25, 27]. Interactive access is provided under http://www.izbi.de.

The paper is organized as follows. In the next section we give an overview of our data integration approach implemented in GenMapper. Section 3 presents the generic data model GAM. Section 4 discusses the data import phase and the generation of annotation views. Section 5 describes additional aspects of the technical implementation as well as an application scenario of GenMapper. Section 6 concludes the paper.

2. Overview of GenMapper

To better understand the problem that GenMapper addresses it is instructive to examine some typical annotation data that is available to the biologist when gathering information about a molecular-biological object of interest. Figure 1 shows annotations for a genetic locus with the source-specific identifier (accession) *353* in the popular public source LocusLink. As indicated in the figure, the locus is annotated by a variety of information from other public sources, e.g., Enzyme [3] for enzyme classification, and OMIM [9] for disease information, and vocabularies and taxonomies such as Hugo [5] for official gene symbols and GeneOntology (GO) [4] for standardized gene functions. GenMapper focuses on combining this kind of inter-related information during data integration and making it directly available for analysis.

Figure 2 shows an overview of the GenMapper integration approach. Integration of source data is performed in two phases: *Data import* and *View generation*. In the first phase, source data is downloaded, parsed and imported into a central relational database following the generic GAM representation. This representation is used for

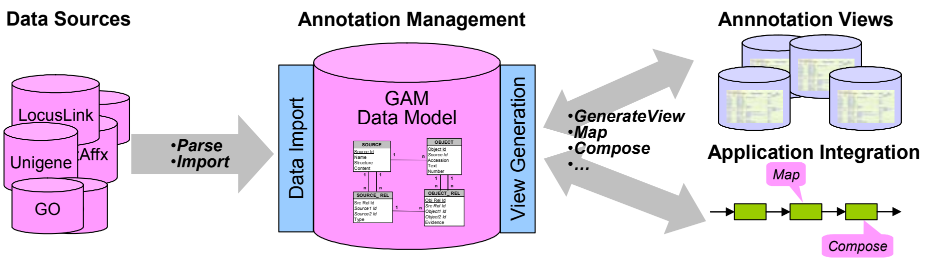
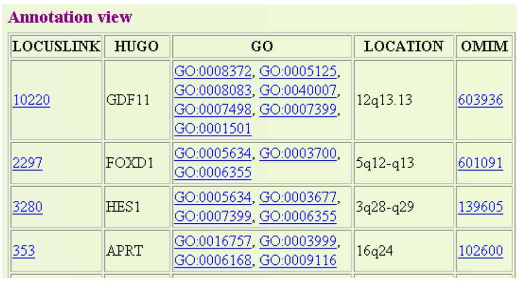


Figure 2. GenMapper architecture for annotation integration

objects and their annotations originating from different sources, such as public sources and taxonomies, as well as the different kinds of relationships.

Since directly accessing the GAM representation may result into complex queries, applications and users are typically provided with annotation views tailored to their analysis needs. Figure 3 shows an example of such an annotation view for some Locuslink genes. In such a view, GenMapper can combine information and annotations from different sources for an arbitrary number of objects. Both the objects (the loci from LocusLink in the example) and the kinds of annotations (e.g., Hugo, GO, Location, and OMIM) can be chosen arbitrarily. Such annotation views are very helpful for comparing and inferring functions of the objects, e.g., if they have been detected to show some correlated behavior in experimental processes.


Annotation view

LOCUSLINK	HUGO	GO	LOCATION	OMIM
10220	GDF11	GO:0008372, GO:0005125, GO:0008083, GO:0040007, GO:0007498, GO:0007399, GO:0001501	12q13.13	603936
2297	FOXD1	GO:0005634, GO:0003700, GO:0006355	5q12-q13	601091
3280	HES1	GO:0005634, GO:0003677, GO:0007399, GO:0006355	3q28-q29	139605
353	APRT	GO:0016757, GO:0003999, GO:0006168, GO:0009116	16q24	102600

Figure 3. An annotation view for LocusLink genes

In general, an annotation view is a structured (e.g., tabular) representation of annotations for objects of a particular source. Annotation views are queryable to support high-volume analysis. A view consists of several attributes which are derived from one source or different sources. The choice of attributes is not fixed as in the underlying sources but can be tailored to application needs. Enabling such a flexible generation of annotation views requires the combination of both objects and annotations, i.e. relationships between objects. This is supported by the uniform representation of data from different sources in our approach.

The annotation views can be flexibly constructed by means of various high-level functions which can operate on entire sources and mappings or a subset of them. Key operators include *Compose* and *GenerateView,* and are specifically defined on the GAM data model. They also represent the means to integrate GenMapper with external applications to provide automatic analysis pipelines with annotation data.

3. The Generic Annotation Model (GAM)

Generic data models aim at uniformly representing different data and metadata for easy extensibility, evolution, and efficient storage. Typically, metadata and data are stored together in triples of object-attribute-value (also coined as Entity-Attribute-Value (EAV) [28]). A molecular-biological example of such a triple is (*APRT*, Name, *adenine phosphoribosyltransferase*). This approach has been used in repository systems to maintain database schemas from different data models [15], in e-Commerce to manage electronic catalogs [13], in the medical domain to manage sparse patient data [28], or in the Semantic Web context to describe and exchange metadata [10].

In GenMapper, we follow the same idea to achieve a generic representation for molecular-biological annotation data by using a generic data model called GAM (Generic Annotation Model). Figure 4 shows the core elements of GAM in a relational format. In particular, we have enriched the EAV representation with several specific properties. First, to avoid the mix of metadata and data in EAV triples and to facilitate data integration from many sources, we explicitly provide two levels of abstraction, *Source* and *Object*. A source may be any predefined set of objects, e.g. a public collection of genes, an ontology, or a database schema. Second, we allow relationships of different semantics and cardinality to be defined at both the source and object level (*Source_Rel* and *Object_Rel*). Both intra- and inter-source relationships are possible. A relationship at the source level (a *mapping*) typically consists of many relationships at the object level (*associations*).

We roughly differentiate between *gene-oriented*, *protein-oriented* and *other* sources according to their content. A source, whose objects are organized in a particular structure, such as a taxonomy or a database schema, is indicated as a *Network* source. Typically, each object has a unique source-specific identifier or accession, which is often accompanied by a textual component, for example to represent the name of the object. Alternatively, an object may also have a numeric representation.

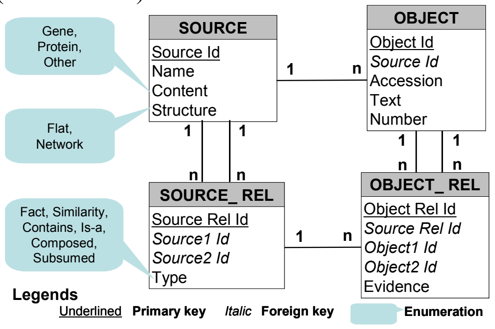


Figure 4. The GAM data model

In *Source_Rel*, we distinguish three types of relationships between and within sources. *Structural* and *annotation* relationships are imported from external data sources and represent the internal structure of a source or semantic correspondences between sources, respectively. In addition, GenMapper supports the calculation and storage of *derived* relationships to increase the annotation knowledge and to support frequent queries. We discuss the single types of relationships in the following.

Annotation relationships. Annotations are determined using different computational or manual methods and typically specified by cross-references between sources. These relationships represent the most important and also the largest amount of data to be managed. Currently we group them into *Fact* and *Similarity* mappings. The former indicate relationships which can be taken as facts, for example, the position of a gene on the genome, while the latter contain computed relationships, e.g. determined by sequence comparisons and alignments (homology) between instances or by an attribute matching algorithm. In *Object_Rel*, an *evidence* value can be captured to indicate the computed plausibility of the association between two any objects.

Structural relationships. Source structure is captured by *Contains* and *IS_A* relationships. *Contains* denotes containment relationships between a source and its partitions, such as between GO and its sub-taxonomies Biological Process, Molecular Function and Cellular Component [4], while *IS_A* is the typical semantic relationship found between terms within a taxonomy like Biological Process or Enzyme.

Derived relationships. Two forms of derived relationships, *Composed* and *Subsumed*, are supported. Composed relationships combine cross-references across several sources to determine annotations that are not directly available. For example, the new mapping Unigene↔GO can be derived by combining two existing mappings, Unigene↔LocusLink and LocusLink↔GO. Subsumed relationships are automatically derived from the *IS_A* structure of a source and contain the associations of a term in a taxonomy to all subsumed terms in the term hierarchy. This is motivated by the fact that if a gene is annotated with a particular GO term, it is often necessary to consider the subsumed terms for more detailed gene functions.

4. Data Integration in GenMapper

In the following we first discuss the data import process. We then outline the use of high-level operators to generate annotation views from the GAM representation.

4.1. Data Import

The integration of new data sources into the GAM data model is performed in two steps, *Parse* and *Import*. For all sources, the output of the *Parse* step is uniformly stored in a simple EAV format as illustrated by the example shown in Table 1 for the locus *353* from Figure 1. It represents a straightforward way to capture annotations as provided on the web pages of public data sources, and therefore makes the construction of parsers very simple.

The *Import* step transforms and integrates data from the EAV into the GAM format. To prevent that already existing sources, objects, mappings and associations are inserted again we perform a duplicate elimination at the source and object level. At the object level we compare object accessions and at the source level we examine source names and audit information, such as date and release of a source. Integrating new data requires relating provided associations with existing data. For example, if GO has already been integrated into GAM, re-importing LocusLink only requires to relate the new LocusLink objects with the existing GO terms.

Table 1. Parsed annotation data from LocusLink

Locus	Target	Accession	Text
353	Hugo	APRT	adenine phosphoribo-syltransferase
353	Location	16q24	
353	Enzyme	2.4.2.7	
353	GO	GO:0009116	nucleoside metabolism
…	…	…	…

The functional split between the *Parse* and *Import* steps helps us to keep the integration effort low. *Parse* represents a small portion of source-specific code to be implemented, while *Import* realizes a generic EAV-to-GAM transformation and migration module and only needs to be implemented once. This makes the integration of a new source relatively easy, mainly consisting of the effort to write a new parser.

4.2. View Generation

To explore the relationships between molecular-biological objects, scientists often have to ask queries in the form "*Given a set of LocusLink genes, identify those that are located at some given cytogenetic positions (Location), **and** annotated with some given GO functions, **but not** associated with some given OMIM diseases*". Such queries exhibit the following properties:

- A query involves one or more mappings between objects of a single *source*, e.g. LocusLink, and one or more *targets* providing the annotations of interest, e.g. Location, GO and OMIM. Both the source and the targets can be confined to subsets of relevant objects.
- The mappings can be used to evaluate logical conditions between objects, i.e. whether they have/do not have some associated annotations. The mappings can be

combined using the logical operators AND or OR and individually negated using the logical operator NOT.

GenMapper supports the specification and processing of such queries by means of tailored annotation views, which can be flexibly constructed using a set of high-level GAM-based operators. In the following, we briefly present some simple operations, such as *Map*, *Range*, and *Domain* (see Table 2), and discuss the most important operations to determine annotations views, *Compose* and *GenerateView*, in more detail. Note that the operations are described declaratively and leave room for optimizations in the implementation.

Simple Operations. The *Map* operation takes as input a source S to be annotated and a target T providing annotations. It searches the database for an existing mapping between S and T and returns the corresponding object associations. *Domain* and *Range* identify the source and the target objects, respectively, involved in a mapping. *RestrictDomain* and *RestrictRange* return a subset of a mapping covering a given set of objects from the source and from the target, respectively.

Table 2. Definitions and examples for some simple operations

Operation	Definition	Example
Map(S, T)	Identify associations between S and T	map = *Map*(S, T) = $\{s_1 \leftrightarrow t_1, s_2 \leftrightarrow t_2\}$
Domain(map)	SELECT DISTINCT S FROM map	*Domain*(map) = $\{s_1, s_2\}$
Range(map)	SELECT DISTINCT T FROM map	*Range*(map) = $\{t_1, t_2\}$
RestrictDomain(map, s)	SELECT * FROM map WHERE S in s	*RestrictDomain*(map, $\{s_1\}$) = $\{s_1 \leftrightarrow t_1\}$
RestrictRange(map, t)	SELECT * FROM map WHERE T in t	*RestrictRange*(map, $\{t_2\}$) = $\{s_2 \leftrightarrow t_2\}$

Compose. The *Compose* operation is based on a simple intuition: transitivity of associations to derive new mappings from existing ones. For example, if a locus l in LocusLink is annotated with some GO terms, so are the Unigene entries associated with locus l. *Compose* takes as input a so-called *mapping path* consisting of two or more mappings connecting two sources with each other, for which a direct mapping is required. For example, it can use a relational join operation to combine map_1: $S_1 \leftrightarrow S_2$ and map_2: $S_2 \leftrightarrow S_3$, which share a common source S_2, and produce as output a mapping between S_1 and S_3.

Compose represents a simple but very effective way to derive new useful mappings. The operation can be used to derive new annotations, which are not directly available in existing sources and their cross-references. However, *Compose* may lead to wrong associations when the transitivity assumption does not hold. This effect can be restricted by allowing *Compose* to be performed with explicit user confirmation on the involved mapping path. The use of mappings containing associations of reduced evidence is a promising subject for future research.

GenerateView. This operation assumes a source S to be annotated and a set of targets T_1, ..., T_m, providing required annotations. The relevant source and target objects are given in the corresponding subsets s and t_1, ..., t_m, respectively, each of which may also cover all existing objects of a source. Finally, the operation requires a method for combining the mappings (AND or OR), and a list of targets for which the obtained mappings are to be negated. The result of such a query is a view of m+1 attributes, S, T_1, …, and T_m, containing tuples of related objects from the corresponding sources. In particular, *GenerateView* implements the pseudo-code shown in Figure 5 to build the required annotation view V.

```
GenerateView(S, s, T_1, t_1, ..., T_m, t_m, [AND|OR], {negated})
V = s                                          //Start with all given source objects
For i = 1..m
    Determine mapping M_i: S↔T_i               //Using either the Map or Compose operation
    m_i = RestrictDomain(M_i, s)               //Consider the given source and target objects
    m_i = RestrictRange(m_i, t_i)
    If negated[T_i]                            //The mapping is specified as negated
        s_i = s \ Domain(m_i)                  //Source objects not involved in the sub-mapping
        m_i = RestrictDomain(M_i, s_i)         //Find associations for these objects
        m_i = m_i right outer join s_i on S    //Preserve objects without associations
    End If
    V = V inner join / left outer join m_i on S    //AND: inner join, OR: left outer join
End For
```

Figure 5. The algorithm for *GenerateView*

V is first set to the given set s of relevant source objects. For each target T_i, a mapping M_i between S and T_i is to be determined. It may already exist in the database, or in many cases, may be not yet available. In the former case, the required mapping is directly retrieved using the *Map* operation. In the latter case, we try to derive such a mapping from the existing ones using the *Compose* operation. A subset m_i is then extracted from M_i to only cover the relevant source objects s and target objects t_i. If necessary, the negation of m_i is built from the subset s_i of s containing the objects not involved in m_i. Finally, V is incrementally extended by performing a left outer join (OR) or inner join (AND) operation with the sub-mapping m_i.

5. Implementation and Use

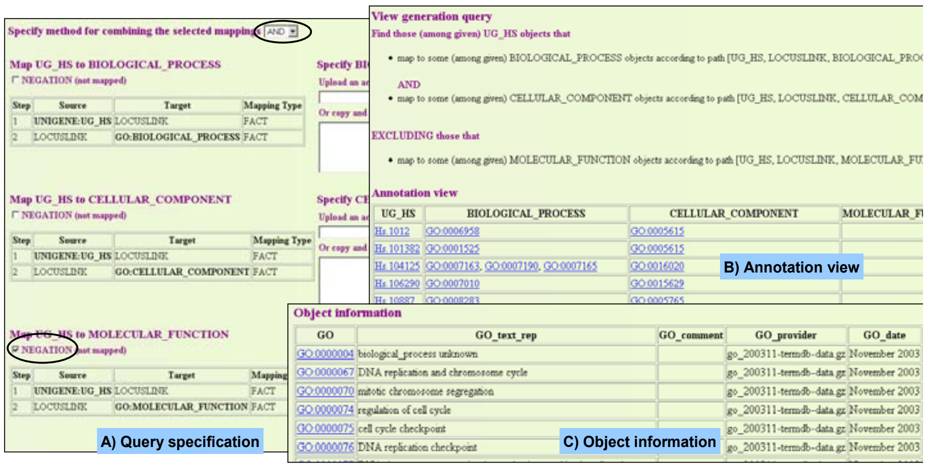
GenMapper is implemented in Java. We use the free relational database management system MySQL to host the backend database implementing the GAM data model. It currently contains approx. 2 million objects of over 60 data sources, and 5 million object associations organized in over 500 different mappings. In the following we present basic functionalities of the interactive user interface and discuss the use of GenMapper in a large-scale analysis application.

5.1. Interactive Query Interface

The interactive interface of GenMapper allows the user to pose queries and retrieve annotations for a set of given objects from a particular source. First, the relevant source can be selected from the list of currently imported sources. The accessions of the objects of interest can be uploaded from a file or manually copied and pasted. If no accessions are specified, the entire source will be considered.

In the next step, the user can specify all targets of interest from the available sources. GenMapper internally manages a graph of all available sources and mappings. Using a shortest path algorithm, GenMapper is able to automatically determine a mapping path to traverse from the source to any specified target. The user can also search in the graph for specific paths, for example, with a particular intermediate source. With a high degree of inter-connectivity between the sources, many paths may be possible. Hence, GenMapper also allows the user to manually build and save a path customized for specific analysis requirements.

When the relevant paths have been selected or manually constructed, the user can specify the target accessions of interest, the method for combining the mappings, and the negation of single mappings as shown in the screenshot in Figure 6a. GenMapper then applies the *GenerateView* operation to construct the annotation view (Figure 6b).



Specify method for combining the selected mappings AND

Map UG_HS to BIOLOGICAL_PROCESS

NEGATION (not mapped)

Step	Source	Target	Mapping Type
1	UNIGENE:UG_HS	LOCUSLINK	FACT
2	LOCUSLINK	GO:BIOLOGICAL_PROCESS	FACT

Map UG_HS to CELLULAR_COMPONENT

NEGATION (not mapped)

Step	Source	Target	Mapping Type
1	UNIGENE:UG_HS	LOCUSLINK	FACT
2	LOCUSLINK	GO:CELLULAR_COMPONENT	FACT

Map UG_HS to MOLECULAR_FUNCTION

[✓ NEGATION] (not mapped)

Step	Source	Target	Mapping
1	UNIGENE:UG_HS	LOCUSLINK	FACT
2	LOCUSLINK	GO:MOLECULAR_FUNCTION	FACT

Specify BI

Upload an a

Or copy and

Specify CE

Upload an a

Or copy and

View generation query

Find those (among given) UG_HS objects that

- map to some (among given) BIOLOGICAL_PROCESS objects according to path [UG_HS, LOCUSLINK, BIOLOGICAL_PRO

AND

- map to some (among given) CELLULAR_COMPONENT objects according to path [UG_HS, LOCUSLINK, CELLULAR_COM

EXCLUDING those that

- map to some (among given) MOLECULAR_FUNCTION objects according to path [UG_HS, LOCUSLINK, MOLECULAR_FU

Annotation view

UG_HS	BIOLOGICAL_PROCESS	CELLULAR_COMPONENT	MOLECULAR_F
Hs.1012	GO:0006958	GO:0005615	
Hs.101382	GO:0001525	GO:0005615	
Hs.104125	GO:0007163, GO:0007190, GO:0007165	GO:0016020	
Hs.106290	GO:0007010	GO:0015629	
Hs.10887	GO:0008283	GO:0005765	

Object information

GO	GO_text_rep	GO_comment	GO_provider	GO_date
GO:0000004	biological_process unknown		go_200311-termdb-data.gz	November 2003
GO:0000067	DNA replication and chromosome cycle		go_200311-termdb-data.gz	November 2003
GO:0000070	mitotic chromosome segregation		go_200311-termdb-data.gz	November 2003
GO:0000074	regulation of cell cycle		go_200311-termdb-data.gz	November 2003
GO:0000075	cell cycle checkpoint		go_200311-termdb-data.gz	November 2003
GO:0000076	DNA replication checkpoint		go_200311-termdb-data.gz	November 2003

Figure 6. Query specification and annotation view for Unigene objects

The interesting accessions among the retrieved ones can be selected to start a new query. Alternatively, the user can retrieve the names and other information of the corresponding objects (Figure 6c). All results can be saved and downloaded in different formats for further analysis in external tools.

5.2. Large-scale Automatic Gene Functional Profiling

In an ongoing cooperation project aiming at a comparative analysis between humans and their closest relatives, chimpanzees [18], GenMapper has been successfully integrated within an automated analysis pipeline to perform complex and large-scale functional profiling of genes.

Gene expression measurements have been performed using Affymetrix microarray technology [1]. From a total of approx. 40.000 genes, the expression of around 20.000 genes were detected, from which around 2.500 show a significantly different expression pattern between the species thus representing candidates for further examination [25, 27]. Functional profiling of the differently expressed genes was based on the analysis of the annotations about their known functions as specified by GeneOntology (GO) terms. In particular, the genes are classified according to the GO function taxonomy in order to identify the functions, which are conserved or have changed between humans and chimpanzees.

Using the mappings provided by GenMapper, the proprietary genes of Affymetrix microarrays were mapped to the generally accepted gene representation UniGene, for which GO annotations were in turn derived from the mappings provided by LocusLink. Furthermore, using the structure information of the sources, i.e. *IS_A* and *Subsumed* relationships, comprehensive statistical analysis over the entire GO taxonomy was possible to determine significant genes. The adopted analysis methodology is also applicable to other taxonomies, e.g. Enzyme, to gain additional insights.

6. Conclusions

We presented the GenMapper system for flexible integration of heterogeneous annotation data. We use a generic data model called GAM to uniformly represent annotations from different sources. We exploit existing associations between objects to drive data integration and combine annotation knowledge from different sources to enhance analysis tasks. From the generic representation we derive tailored annotation views to serve specific analysis needs and queries. Such views are flexibly constructed using a set of powerful high-level operators, e.g. to combine annotations imported from different sources. GenMapper is fully operational, integrates data from many sources and is currently used by biologists for large-scale functional profiling of genes.

Acknowledgements. We thank Phil Bernstein, Sergey Melnik and Peter Mork and the anonymous reviewers for helpful comments. This work is supported by DFG grant BIZ 6/1-1.

References

1. Affymetrix: http://www.affymetrix.com/
2. Ensembl: http://www.ensembl.org/
3. Enzyme: http://www.expasy.ch/enzyme/
4. GeneOntology: http://www.geneontology.org/
5. Hugo: http://www.gene.ucl.ac.uk/nomenclature/
6. Human Genome Browser: http://genome.ucsc.edu/
7. InterPro: http://www.ebi.ac.uk/interpro/
8. LocusLink: http://www.ncbi.nlm.nih.gov/LocusLink/
9. OMIM: http://www.ncbi.nlm.nih.gov/omim/
10. RDF: http://www.w3.org/RDF/
11. SwissProt: http://www.expasy.ch/sprot/
12. Unigene: http://www.ncbi.nlm.nih.gov/UniGene/
13. Agrawal, R., A. Somani, Y. Xu: Storage and Querying of E-Commerce Data. Proc. VLDB, 2001
14. Baxevanis, A.: The Molecular Biology Database Collection: 2003 Update. Nucleic Acids Research 31(1), 2003
15. Bernstein, P. et al.: The Microsoft Repository. Proc. VLDB, 1997
16. Critchlow, T. et al.: DataFoundry: Information Managemenet for Scientific Data. IEEE Trans. on Information Management in Biomedicine 4(1), 2000
17. Dowell, R.D. et al.: The Distributed Annotation System. BMC Bioinformatics 2(7), 2001
18. Enard, W. et al.: Intra- and Inter-specific Variation in Primate Gene Expression Patterns. Science 296, 2002
19. Etzold, T., A. Ulyanov, P. Argos: SRS – Information Retrieval System for Molecular Biology Data Banks. Methods in Enzymology 266, 1996
20. Fujibuchi, W. et al.: DBGET/LinkDB: An Integrated Database Retrieval System. Proc. PSB, 1997
21. Goble, C. et al.: Transparent Access to Multiple Bioinformatics Information Sources. IBM System Journal 40(2), 2001
22. Haas, L. et al.: DiscoveryLink – A System for Integrated Access to Life Sciences Data Sources, IBM System Journal 40(2), 2001
23. Kementsietsidis, A., M. Arenas, R.J. Miller: Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues. Proc. SIGMOD, 2003
24. Kemp, G., N. Angelopoulos, P. Gray : A Schema-based Approach to Building Bioinformatics Database Federation. Proc. BIBE, 2000
25. Khaitovich, P., et al.: Evolution of Gene Expression in the human brain. Submitted for publication
26. Lacroix, Z., T. Critchlow (Ed.): Bioinformatics: Managing Scientific Data, Morgan Kaufmann, 2003
27. Mützel, B., H.-H. Do, P. Khaitovich, P., G. Weiß, E. Rahm, S. Pääbo: Functional Profiling of Genes Differently Expressed in the Brains of Humans and Chimpanzees. In preparation
28. Nadkarni, P. et al.: Organization of Heterogeneous Scientific Data Using the EAV/CR Representation. Journal of American Medical Informatics Association 6(6), 1999
29. Paton, N. et al.: Conceptual Modeling of Genomic Information. Bioinformatics 16(6), 2000
30. Ritter, O.: The Integrated Genomic Database (IGD). In Suhai, S. (Ed.): Computational Methods in Genome Research. Plenum Press, 1994
31. Wong, L.: Kleisli, Its Exchange Format, Supporting Tools, and an Application in Protein Interaction Extraction. Proc. BIBE, 2000