A Tool for Evaluating Strategies for Grouping of Biological Data

Vaida Jakonienė, Patrick Lambrix

Department of Computer and Information Science
Linköpings universitet, SE-581 83 Linköping, Sweden

Summary

During the last decade an enormous amount of biological data has been generated and techniques and tools to analyze this data have been developed. Many of these tools use some form of grouping and are used in, for instance, data integration, data cleaning, prediction of protein functionality, and correlation of genes based on microarray data. A number of aspects influence the quality of the grouping results: the data sources, the grouping attributes and the algorithms implementing the grouping procedure. Many methods exist, but it is often not clear which methods perform best for which grouping tasks. The study of the properties, and the evaluation and the comparison of the different aspects that influence the quality of the grouping results, would give us valuable insight in how the grouping procedures could be used in the best way. It would also lead to recommendations on how to improve the current procedures and develop new procedures. To be able to perform such studies and evaluations we need environments that allow us to compare and evaluate different grouping strategies. In this paper we present a framework, KitEGA[1], for such an environment, and present its current prototype implementation. We illustrate its use by comparing grouping strategies for classifying proteins regarding biological function and isozymes.

1 Introduction

During the last decade an enormous amount of biological data has been generated and techniques and tools to analyze this data have been developed. Many of these tools use techniques to group data representing similar entities. For instance, data clustering and classification techniques are used to find similar sequences for predicting the functionality of new sequences (e.g. [9]), to find correlated genes based on microarray data (e.g. [8, 19, 20, 22]), or to classify publications according to an ontology to locate relevant documents faster (e.g. [7]). Grouping of data entries in one or more data sources is also an operation underlying many different data management tasks. Grouping can be used to structure and visualize search results in a convenient way for the user. The identification of similar data entries and their grouping are also core operations when performing data cleaning activities (e.g. [14]). In the context of data integration, techniques underlying grouping are important to correlate data entries at different data sources. Also duplicate detection, which is used for data cleaning (e.g. [16]) and for data integration (e.g. [21, 2]), can be seen as a grouping task where it is required that grouped data entries represent the same real-world object.

Many of the approaches for grouping are based on the computation of a similarity value (or equivalently, a distance measure) between objects. Different techniques are developed to compute a similarity value between objects based on the object types. For instance, edit distance and

[1] Tool*Kit* for *E*valuation of *G*rouping *A*lgorithms, http://www.ida.liu.se/∼iislab/projects/KitEGA/

n-gram are well-established techniques to define similarity between strings, while BLAST can be used to define a similarity measure between DNA or protein sequences. Recently, a number of projects discussed methods to compute semantic similarity over terms in a Gene Ontology (GO) [11] ontology (e.g. [5, 23]). The similarity between GO terms can be used to compute a similarity between data entries that are annotated with these GO terms (e.g. [17, 15]).

Data entries in biological data sources are often complex and store different types of information. Although most of the research has focused on organizing the data based on aspects, such as sequence similarity and function, we need to analyze data using different aspects and from different points of view to obtain deeper insights in the characteristics of the data and to discover new knowledge. This means that we need to be able to organize the data based on different attributes or different combinations of attributes. In the following we use the term *grouping* to refer to the task of organizing the data according to a certain attribute or a combination of attributes. Further, we concentrate on the task of *similarity-based grouping*. During similarity-based grouping the analyzed data entries are compared with respect to a selected subset of attributes, and similarity functions that are relevant to the attributes are used to compute the similarity of the stored values.

A number of aspects influence the quality of the grouping results: the data sources, the grouping attributes and the algorithms implementing the grouping procedure. In some cases, for a given grouping task, it can be difficult to decide on which attributes to perform grouping. Also, different sets of attributes may seem relevant to the grouping task, but lead to varying quality of the results [16]. Further, suitable algorithms need to be selected to compute the similarity between data entries and to organize similar data entries into groups. Many methods exist, but it is often not clear which methods perform best for which grouping tasks. The study of the properties, and the evaluation and the comparison of the different aspects that influence the quality of the grouping results, would give us valuable insight into the best way to use the grouping procedures. It would also lead to recommendations on how to improve the current procedures and develop new procedures. Although some evaluations have been performed, this has often been a cumbersome task where many components needed to be re-implemented (e.g. section 2.2). To be able to perform such studies and evaluations in a more efficient way we need environments that allow us to compare and evaluate different grouping procedures.

In this paper we present such an environment (KitEGA). KitEGA is based on the method proposed in [15]. It allows evaluators to plug in their own algorithms related to the grouping strategies and the evaluation measures, as well as their own data sets, and provides support for the analysis of the grouping results. The method from [15] and other background is discussed in section 2. We present KitEGA and its current prototype implementation in section 3. In section 4 we exemplify the use of KitEGA by comparing grouping strategies for classifying proteins regarding biological function and isozymes. The paper concludes in section 5.

2 Background

2.1 Method for grouping biological data

In [15] a method that supports similarity-based grouping of biological data and that enables the development of grouping procedures was proposed. It covers the steps proposed for general

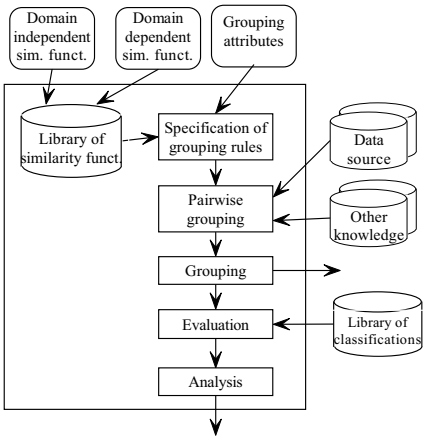

Figure 1: Method for similarity-based grouping [15].

clustering in [13] and duplicate detection in [16]. The components and the main steps of the method are illustrated in figure 1. The method uses as input the data source on which the grouping is performed. We note that before the grouping procedure can be applied to a data source, the data source usually needs to go through a number of data transformation steps, such as merging of data and data translation from one format to another. Further, the method uses domain-dependent and domain-independent similarity functions that can compute similarity values between data values, and grouping attributes on which we base the computation of the similarity of data entries in the data source. Also external sources may support the grouping task. Based on this input the method can generate groupings of data. In addition, the method also allows the evaluation and analysis of the grouping results. For this purpose we use a library of known classifications. The library stores selected sets of data entries organized into classes[2]. The method returns the generated groups as well as reports from the evaluation and analysis.

The main steps in the method are the following. In our method grouping rules are used to express conditions on which two data entries are compared for similarity. During *specification of grouping rules* the user defines a grouping rule or selects an already available grouping rule that is deemed to be relevant to the current grouping task. A grouping rule may combine different similarity functions applied to one or more grouping attributes. During the *pairwise grouping* step the similarity between pairs of data entries are computed. Auxiliary domain knowledge may be used. The result of this step is the identification of the pairs of data entries that are similar. The pairwise grouping includes the following sub-steps: a) selection of pairs of data entries to be compared; b) comparison of data values of the selected grouping attributes by applying the similarity functions; and c) comparison of the selected data entries on the basis of given grouping rules. While for small data sources all pairs of data entries can be analyzed, for large data sources pruning techniques may be used to decrease the number of performed comparisons. The *grouping* step takes as input pairs of similar data entries and organizes the data entries into a set of groups composed of similar data entries. Different techniques can be used to perform grouping and they may vary on a number of aspects. For instance, the groups can be allowed to overlap or may be required to be disjoint. Some approaches may require the transitivity property between similar data entries or they may allow to ignore some similarity relationships, e.g. in order to split a group into smaller groups. During the *evaluation*

[2]In the rest of the paper we use *classes* to refer to given classifications and *groups* to refer to the results of grouping techniques.

step different measures are computed to evaluate the quality of the grouping results. We can distinguish between two kinds of quality measures (e.g [13, 12]): internal and external. Internal quality measures compare different groupings based only on information obtained during the grouping (e.g. pairwise similarity between data entries). External quality measures evaluate the grouping results with respect to known classes. Each validation technique has its own bias and different strengths and weaknesses, and often we will want to compare the grouping procedures with respect to a number of measures. In general, when performing grouping validation for the evaluation and comparison of algorithms, we primarily want to use external measures. When focusing on grouping validation for a novel data set, internal measures are used [13]. In our method, the library of classifications is used to compute external quality measures. Finally, during the *analysis* step the grouping and evaluation results are analyzed. Different forms and reports are generated providing support for exploring the results from different points of view. For instance, valuable insight is gained by analyzing and studying the entries belonging to a single group, the correlation between groups and classes, and the influence of external knowledge on the results.

2.2 Evaluation of grouping methods

A number of evaluations of different kinds of grouping algorithms have been performed. For instance, regarding clustering of gene expression data [24] proposes a measure to estimate the predictive power of a clustering algorithm and compares 2 partitional and 3 hierarchical clustering algorithms based on this measure. [6] proposes 3 validation strategies and compares 6 algorithms. Also [10] proposes a new validation measure and compares 4 clustering methods. 5 biclustering methods for gene expression data are evaluated in [18]. Common to all these evaluations is the fact that they focus on cluster validation for the evaluation and comparison of algorithms. They use synthetic and real data sources. Some of the papers also aim to propose new validation measures. Further, in all these evaluations, most of the evaluated algorithms needed to be re-implemented for the purpose of the evaluations. [4] presents the SecondString Toolkit (http://secondstring.sourgeforge.net) for name-matching methods which could be used, for instance, in duplicate detection. Several distance functions for strings are implemented. The algorithms are compared on a data set regarding non-interpolated average precision. A system that goes some way into providing an *environment* for clustering and validation is the Machaon Cluster Validation Environment [3]. This system is intended for clustering of microarray data and evaluating the quality of the obtained clusters. The system focuses on cluster validation for new data sets and therefore uses internal measures based on compactness and isolation. The system implements several clustering algorithms, metrics (distance), and internal measures. The user can choose among these to run a cluster task on a data set. The results are shown as a tree. The highest level nodes represent the chosen cluster algorithms with particular parameter selection. The next level represents the results of applying different validity measures to the clusters generated by the algorithm.

The framework and system (KitEGA) that we propose in the next section aims to go one step further. KitEGA is a platform for evaluating and comparing similarity-based grouping strategies. Evaluators can plug in their own algorithms related to the grouping strategies and the evaluation measures, as well as their own data sets. KitEGA provides then support for running the algorithms, and summarizing and analyzing the results.

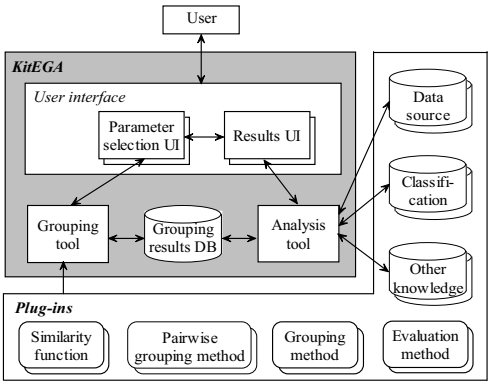

Figure 2: The KitEGA framework.

3 KitEGA

In this section we present the KitEGA framework for evaluating grouping strategies and its current prototype implementation. The framework realizes the similarity-based grouping method discussed in the previous section.

3.1 Framework

Figure 2 illustrates the KitEGA framework. KitEGA receives as an input a set of components (plug-ins) that define the grouping procedures that we want to evaluate. The plug-ins include alternative implementations of the steps defined in the grouping method, i.e. similarity functions, and pairwise grouping, grouping and evaluation methods. Also data sources on which to perform grouping, auxiliary domain knowledge and classifications are provided to the system as plug-ins and can be loaded and used on demand. In addition, the framework takes as an input user selected parameters that specify a studied grouping task (test case). As output the system returns grouping, evaluation and analysis results presented as forms and reports.

The framework provides a number of user interfaces (UI) enabling user interaction with the system. To define a test case, a group of interfaces is used that informs the user about the possible choices and accepts parameters specifying the user's choice (figure 2, parameter selection UI). For instance, the parameters specifying a test case may define the selection of a data source, the specification of a new grouping rule or the selection of a predefined grouping rule, the selection of pairwise grouping, grouping and evaluation methods, and the presentation of the results. In addition, the framework provides a number of UIs that provide different ways to view the results of a study (figure 2, results UI). The study may be an analysis of a single test case or a comparative analysis of a number of test cases.

The grouping tool in the framework is responsible for executing a test case. The tool has access to the loaded components and receives as input parameters specifying the test case. During the test case execution the grouping tool runs the selected pairwise grouping, grouping and evaluation methods. The grouping and evaluation results are saved into a database. The grouping results database can be used to store results for several test cases. The analysis tool acts as a mediator between the results UI and the different types of data available for the framework. The analysis tool collects the relevant data. This includes the retrieval and integration of data from

the grouping results database, the used data sources, the classification and auxiliary knowledge. The data may also be post-processed, for instance to compute the number of entries in each group, or to find false positives in a group with respect to a class.

3.2 Implementation

In the current prototype implementation of KitEGA we focused on the flexible specification of test cases enabled through the use of plug-ins and on the development of user interfaces supporting the specification of test cases and the representation of the grouping results. This has led to a number of design choices. The current implementation covers all parts described in the framework except that it does not allow the user to interactively specify the grouping rules. Instead, for this first prototype, we require that the grouping rules are loaded into the system as a plug-in. As a result, the user selects a grouping rule for a test case from a list of given rules. Further, for our test cases we decided to compute the similarity values between all pairs of data entries and therefore did not need alternative pairwise grouping methods. We also decided to load all the data and knowledge into main memory and not use any database facilities. This was sufficient for the tests that we performed.

The user starts the evaluation process by choosing a number of parameters specifying a test case (figure 3). She selects a data source, a grouping rule, a grouping method, evaluation methods and a classification source to use. The content of this user interface is generated dynamically based on the configuration file specifying the plug-ins that were made available to the system. Further, the user specifies attributes and the maximum size of the data values for the attributes which should be presented in the results (not shown in figure).

KitEGA will then run the test case and present the results to the user. Figure 4 shows the main form presenting grouping and evaluation results for a selected test case. The form shows the data entries included in each group together with information about the class they belong to according to the previously selected classification. Further, the form presents the values of the computed evaluation measures together with some additional information about the test case, e.g. the total number of data entries in the data source and the total number of classes in the classification. The form provides support for starting new test cases, for saving test cases and their results and for loading previously saved test cases. In addition to the basic grouping result, the current KitEGA implementation supports several other forms giving different views on the data. To give a deeper insight into the grouping results, for a selected test case, KitEGA shows how the data entries in the generated groups are distributed among the classes in the selected classification (figure 5). In the figure a row represents a group while a column represents a class. The numbers in parentheses represent the total number of data entries in a group or a class, respectively. A cell stores data on the number of data entries that are true positives (they belong to the group and the class), false positives (they belong to the group, but not to the class) and false negatives (they belong to the class, but not to the group), respectively. Further, the system allows for a given group and class to view detailed information on the true positives, false positives and false negatives (e.g. figure 6). True positives are color-coded. To support comparative analysis of grouping procedures, the system also presents a form gathering evaluation results from different saved test cases (figure 7).

KitEGA is implemented in Java. JavaServer Pages technology is used to build the web-based user interface. The data sources referred to in the KitEGA implementation are Java objects

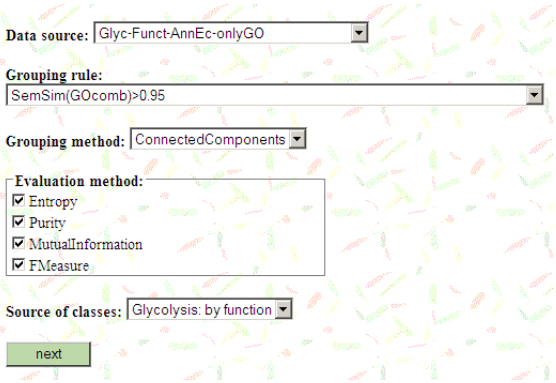

Figure 3: Specification of a test case.

Figure 4: Grouping and evaluation results of a test case.

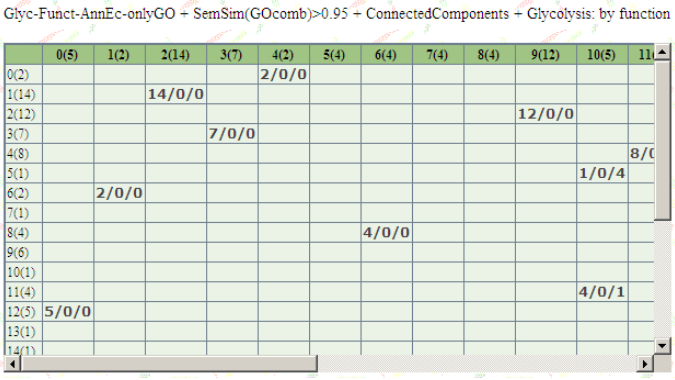

Figure 5: Detailed comparison of groups and classes. (Rows represent groups. Columns represent classes. Numbers in parentheses represent the total number of data entries in a group or a class. A cell stores data on true positives/false positives/false negatives.)

group: 11(4) + class:10(5) + 4/0/1

GroupNr	ClassNr	ID	Definition	GO combined
11	10	P08559	Pyruvate dehydrogenase E1 component alpha subunit, somatic form, mitochondrial precursor (PDHE1-A ty	go:0004739
11	10	NP_000275	pyruvate dehydrogenase (lipoamide) alpha 1 [Homo sapiens].	go:0016491, go:0004739, go:0016624
11	10	P11177	Pyruvate dehydrogenase E1 component beta subunit, mitochondrial precursor (PDHE1-B).	go:0004739
11	10	P29803	Pyruvate dehydrogenase E1 component alpha subunit, testis-specific form, mitochondrial precursor (PD	go:0004739
5	10	P10515	Dihydrolipoyllysine-residue acetyltransferase component of pyruvate dehydrogenase complex, mitochond	go:0004742

Figure 6: Details on true positives/false positives/false negatives for a given group and class.

ID	DataSource	Rule	GrMethod	Classif	# of entries	# of groups	# of classes	Entropy	Purity	MutualInformation	FMeasure
1	Glyc-Funct-Ann-onlyGO	SemSim (GOann) >0.95	ConnectedComponents	Glycolysis: by function	67	26	23	1.0	1.0	0.911770972962663	0.974650556740109
2	Glyc-Funct-AnnSw-onlyGO	SemSim (GOcomb) >0.95	ConnectedComponents	Glycolysis: by function	75	23	24	0.8652654637823463	0.8	0.7942395602417653	0.7917895141895143
3	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.95	ConnectedComponents	Glycolysis: by function	92	26	25	1.0	1.0	0.8810530832230523	0.9939613526570048
4	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.85	ConnectedComponents	Glycolysis: by function	92	21	25	0.777556968313313	0.6956521739130435	0.6824607500357851	0.7074322974655634
5	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.95	Cliques	Glycolysis: by function	92	29	25	1.0	1.0	0.8839495868521302	0.8392424467190822
6	Glyc-Funct-AnnEc-onlyGO	SeqSim(seq) >0.85	ConnectedComponents	Glycolysis: by function	92	41	25	1.0	1.0	0.8231661542255041	0.8046256946714613
7	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.95	ConnectedComponents	Glycolysis: isozymes	92	26	47	0.7921820789964245	0.5869565217391305	0.7969682196233567	0.6484704432358892
8	Glyc-Funct-AnnEc-onlyGO	SeqSim(seq) >0.85	ConnectedComponents	Glycolysis: isozymes	92	41	47	0.9256079005200991	0.8478260869565217	0.8848110696286725	0.8380873956960911

Figure 7: Evaluation result for the saved test cases.

storing the data entries compliant to the KitEGA internal representation of the data.

4 Example use

In this section we illustrate how KitEGA can be used to analyze and compare similarity-based strategies for grouping of biological data, and to gain deeper knowledge about a data source and about the suitability of different grouping strategies for different grouping tasks in a convenient way. We use the test cases proposed in [15] and re-evaluate these using KitEGA. For a more detailed description of the test cases we refer to [15]. We also note that the overhead of storing the results of the running of the grouping procedures is minimal. The response of the system during the analysis phase is immediate.

Set-up for test cases. The test cases cover two grouping tasks: grouping of proteins with respect to their biological function and with respect to what classes of isozymes[3] they belong to. For the analysis we have used a set of data entries including human proteins involved in glycolysis that we retrieved through the Entrez retrieval system. The full data set contains 190 entries. To generate a data source, the data entries in the genpept format were translated into the format supported by KitEGA. For the parsing of the data, we have used the BioJava library. We performed grouping on the attributes SEQUENCE, GO_{ann} and GO_{comb}. SEQUENCE[4]

[3]Proteins are isozymes (or isoenzymes) if they are enzymes that catalyze the same chemical reaction, but they may differ in their amino acid sequences [1]. Isozymes differ in their kinetic properties, the way they are regulated by other proteins and quantities in which they are expressed in different tissues.

[4]In the original file this attribute is called ORIGIN, but for the sake of readability we use the term SEQUENCE.

is the attribute where the amino acid sequence of a protein is stored. GO_{ann} is an attribute that gathers GO terms found in the data entries. For our experiments we found GO terms in the attribute DBSOURCE for data entries originating from SWISS-PROT and in the "note" property in the "CDS" field under the FEATURES attribute for the other data entries. GO_{comb} is an extra attribute that extends a set of GO terms found in GO_{ann} by GO terms that we could generate based on mappings between data values and ontological terms found on the web pages of the GO Consortium. This allows to increase the number of data entries annotated with GO terms. We used the mapping *spkw2go* to translate values of the KEYWORDS attribute into GO terms. Also, we used the mapping *ec2go* to translate values of the EC-NUMBER attribute into GO terms. Different ways to generate values for GO_{comb} resulted in variants of our original data source. As a result, the number of analyzed data entries differs among the data sources. In the evaluations below, we refer to the data sources as Glyc-Funct-Ann-onlyGO, Glyc-Funct-AnnSw-onlyGO and Glyc-Funct-AnnEc-onlyGO. They contain 67, 75 and 92 data entries respectively. The used GO terms originate from only GO_{ann}, GO_{ann} and *spkw2go* mappings, and GO_{ann} and *ec2go* mappings, respectively. The data sources include only the data entries having GO terms. The data entries include only GO terms belonging to the GO function ontology.

In addition to the data sources, for the discussed test cases KitEGA loads the following plug-ins. 1) Classifications of data entries according to biological function and classes of isozymes. The full data set of 190 data entries was manually classified and resulted in 28 disjoint classes for function and 52 disjoint classes for isozymes. 2) Sequence and semantic similarity functions. The sequence similarity function *SeqSim* is a function that performs pairwise sequence alignment and returns a similarity score between sequences. We use a sequence alignment tool implemented in Java, JAligner (http://jaligner.sourceforge.net/). The similarity score is defined as the number of matches in the alignment divided by the length of the alignment. The semantic similarity function *SemSim* is a function that computes the similarity between two sets of GO terms. The function uses an edge-based algorithm to evaluate the distance between two GO terms and treats two sets of GO terms as similar if each term of one set is similar to a term in the other set. The similarity functions in figure 7 are shown with one argument representing the type of the compared values. 3) A set of predefined grouping rules that explore different aspects of the grouping strategies. 4) Two grouping approaches: cliques and connected components. *Cliques* require that all data entries in a group are similar to each other. In this approach, the generated groups may overlap. *Connected components* collect all data entries that are directly or transitively similar to each other into a single group. As a result, the approach generates disjoint groups. 5) External evaluation measures: purity, F-measure, entropy and mutual information. Purity evaluates the average precision of the groups with respect to their best matching classes. F-measure combines precision and recall of the classes with respect to their best matching groups on average. Normalized entropy analyzes how on average the data entries in each group distribute among the classes. Mutual information is a measure of correspondence on average between each group and class.

Illustration of use of KitEGA. To study one particular test case, we can use KitEGA forms such as the ones shown in figures 4, 5 and 6. The form in figure 4 shows all data entries with the group and class they belong to for test case 3 in figure 7. The form in figure 5 shows a detailed comparison between groups and classes for this test case in a way that supports locating the discrepancies and overlaps between the groups and classes. If there is a one-to-one correspondence between groups and classes, then each row and each column only contain one

entry (which then is of the form x/0/0). Several entries in one column show that the data entries in a class are distributed over several groups. Similarly, several entries in a row show that the data entries in a group are distributed over several classes. The form in figure 5 shows that for test case 3 there is a one-to-one correspondence between the groups and the classes except for the distribution of the 5 data entries of class 10 between the groups 5 and 11. This leads us to an interesting point for further study. The form in figure 6 shows details about group 11 and class 10. As we already know from figure 5 there are 4 true positives, no false positives and 1 false negative. This means that the data entries of class 10 were divided into two groups. Figure 6 shows that P11177, NP_000275, P08559 and P29803 are grouped together, while P10515 appears in a separate group. The grouping result can be explained by the fact that class 10 describes an enzyme complex that consists of multiple copies of the three types of enzymes E_1, E_2 and E_3. The goal of the whole enzyme complex is to build the molecule acetyl-CoA from Pyruvate and CoA, but different types of enzymes belonging to the complex vary in their function. In our case, P11177, NP_000275, P08559 and P29803 describe E_1, while P10515 describes E_2. The difference between functions is reflected in the available GO annotations (shown in the form in figure 6): P11177, NP_000275, P08559 and P29803 are annotated with "pyruvate dehydrogenase (acetyl-transferring) activity", i.e. go:0004739, while P10515 has "dihydrolipoyllysine-residue acetyltransferase activity", i.e. go:0004742. These two terms are not closely related in GO.

Figure 7 shows all test cases and presents a summary of the grouping results. When comparing different grouping procedures, this is the natural place to start. A first way to use the summary results is to find the *best test cases* for a particular data source. The results in figure 7 reveal that for the data entries in Glyc-Funct-AnnEc-onlyGO and grouping on function (test cases 3-6) the best results are obtained using grouping on GO_{comb} (test case 3). For Glyc-Funct-AnnEc-onlyGO and grouping on isozymes (test cases 7-8) the best results are obtained using grouping on SEQUENCE (test case 8). It also shows that grouping on sequences is too specific for grouping on function. Test case 6 has nearly twice as many groups as classes.

KitEGA also allows the study of test case components for *different data sources*. By comparing test cases 1-3 in the form in figure 7 we may conclude that *spkw2go* mappings are not suitable for grouping on function. SWISS-PROT keywords are quite general and are mapped to high level GO terms. For instance, some SWISS-PROT data entries contain 'Glycolysis' as a keyword (found in forms similar to figures 4 and 6), while all the data entries in the data source relate to 'Glycolysis'. Therefore, some data entries were grouped together even though they differed in more specific functions, i.e. belonged to different classes. GO terms obtained through *ec2go* mappings, however, were specific enough. For instance, EC:2.7.1.11 maps to the GO term '6-phosphofructokinase activity', which is a very specific function.

The form in figure 7 can also be used to evaluate the impact of *different thresholds* as part of grouping rules. Test cases 3 and 4 show the use of different thresholds (0.95 and 0.85) on the same data source. The results show that a large part of the data entries in the same group are highly similar to some of the other data entries in the group. In general, experiments with different thresholds allow exploration of the correlation of data entries at different levels of similarity.

Test cases 3 and 5 in figure 7 illustrate the impact of the *different grouping approaches*, in our case, connected components and cliques. As cliques put stronger requirements on the grouped data entries and allow overlapping groups, a larger number of groups are generated and a lower

F-measure is obtained. Based on the measures, however, we cannot make a decisive claim about which of the two grouping approaches performs better. The nature of the approaches is very different. They complement each other and give different ways of presenting the results. For instance, by comparing the results of the grouping approaches, subgroups of data entries that are interconnected in a stronger way to each other than to the rest of the entries in the group, can be located (using forms similar to the forms in figures 4, 5 and 6). Such subgroups could be generated for several reasons, such as the fact that the described sequences may slightly differ in functionality or that the data entries may have incomplete information.

5 Conclusion

In this paper we motivated the need for environments that support the evaluation of similarity-based grouping procedures. We presented such an environment and illustrated its use. We intend to extend the KitEGA implementation in several ways. We will extend the system to fully comply with our framework. Further, we will provide a number of libraries for components that are common. This could include, for instance, different evaluation measures or grouping methods. We will also use KitEGA for studies in data integration.

Acknowledgements We thank David Rundqvist for work on the preparation of the test cases. This research work was funded by CUGS (the Swedish national graduate school in computer science) and CENIIT (Center for Industrial Information Technology). The authors are members of the EU NoE REWERSE (FP6 project 506779, WG on a Semantic Web for Bioinformatics).

References

[1] Berg J, Tymoczko J, Stryer L, *Biochemistry*. W.H. Freeman and Company, New York, 2002.

[2] Bilke A, Bleiholder J, Böhm C, Draba K, Naumann F, Automatic Data Fusion with HumMer, Demo at *31st International Conference on Very Large Data Bases*, 1251-1254, 2005.

[3] Bolshakova N, Azuaje F, Cunningham P, An integrated tool for microarray data clustering and cluster validity assessment, *Bioinformatics*, 21(4):451-455, 2005.

[4] Cohen W, Ravikumar P, Fienberg S, A comparison of string metrics for matching names and records, *KDD Workshop on Data Cleaning and Object Consolidation*, 2003.

[5] Couto F, Silva M, Coutinho P, Semantic similarity over the gene ontology: family correlation and selecting disjunctive ancestors, *Conference on Information and Knowledge Management*, 343-344, 2005.

[6] Datta S, Datta S, Comparisons and validation of statistical clustering techniques for microarray gene expression data, *Bioinformatics*, 19(4):459-466, 2003.

[7] Doms A, Schroeder M, GoPubMed: Exploring PubMed with the GeneOntology. *Nucleic Acids Research*, 33:W783-W786, 2005.

[8] Eisen M, Spellman P, Brown P, Botstein D, Cluster analysis and display of genome-wide expression patterns, *PNAS*, 95:14863-14868, 1998.

[9] Gabaldon T, Huynen M, Prediction of protein function and pathways in the genome era, *Cellular and molecular life sciences*, 61(7-8):930-944, 2004.

[10] Gat-Viks I, Sharan R, Shamir R, Scoring clustering solutions by their biological relevance, *Bioinformatics*, 19(18):2381-2389, 2003.

[11] Gene Ontology Consortium, Gene Ontology: tool for the unification of biology, *Nature Genetics*, 25(1):25-29, 2000. http://www.geneontology.org/.

[12] Halkidi M, Batistakis Y, Vazirgiannis M, On clustering validation techniques, *Journal of Intelligent Information Systems*, 17(2-3):107-145, 2001.

[13] Handl J, Knowles J, Kell, D, Computational cluster validation in post-genomic data analysis, *Bioinformatics*, 21(15):3201-3212, 2005.

[14] Herbert K, Gehani N, Piel W, Wang J, Wu C, BIO-AJAX: An Extensible Framework for Biological Data Cleaning, *SIGMOD Record*, 33(2):51-57, 2004.

[15] Jakonienė V, Rundqvist D, Lambrix P, A method for similarity-based grouping of biological data, *3rd International Workshop on Data Integration in the Life Sciences*, LNBI 4075, 136-151, 2006.

[16] Koh J, Lee M, Khan A, Tan P, Brusic V, Duplicate Detection in Biological Data using Association Rule Mining, *ECML/PKDD Workshop on Data Mining and Text Mining for Bioinformatics*, 31-37, 2004.

[17] Lord P, Stevens R, Brass A, Goble C, Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation, *Bioinformatics*, 19(10):1275-1283, 2003.

[18] Prelić A, Bleuler S, Zimmermann Ph, Wille A, Bühlmann P, Gruissem W, Hennig L, Thiele L, Zitzler E, A systematic comparison and evaluation of biclustering methods for gene expression, *Bioinformatics*, 22(9):1122-1129, 2006.

[19] Quackenbush J, Computational genetics: computational analysis of microarray data, *Nature Reviews Genetics*, 2:418-427, 2001.

[20] Shamir R, Sharan R, Algorithmic approaches to clustering gene expression data, *Current Topics in Computational Molecular Biology*, Jiang, Smith, Xu, Zhang (eds), MIT Press, 269-300, 2002.

[21] Schallehn E, Sattler K-U, Saake G, Efficient similarity-based operations for data integration, *Data & Knowledge Engineering*, 48:361-387, 2004.

[22] Slonim D, From patterns to pathways: gene expression data analysis comes of age, *Nature Genetics*, 32:502-508, 2002.

[23] Speer N, Fröhlich H, Spieth C, Zell A, Functional Distances for Genes Based on GO Feature Maps and their Application to Clustering, *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 142-149, 2005.

[24] Yeung K, Haynor D, Ruzzo W, Validating clustering for gene expression data, *Bioinformatics*, 17(4):309-318, 2001.