ISSN:0975-9646

A Genetic Programming Approach for Record Deduplication

[1]R.Parimala devi , [2]Dr. V.Thigarasu

[1]Research Scholar, Department of Computer Science,
Karpagam University, Coimbatore.
[2]Associate Professor, Department of Computer Science,
Gobi Arts and Science College, Gobichettipalayam.

Abstract: Genetics and Molecular Biology are keys for the understanding the mechanisms of many of the human diseases that have strong harmful effects. The empirical mission of Genetics is to translate these mechanisms into Clinical benefits, thus bridging in-silico findings to patient bed side: approaching this goal means achieving what is commonly referred as clinical genomics or personalized medicine. Several systems that rely on the integrity of the data in order to offer high quality services, such as digital libraries and ecommerce brokers, may be affected by the existence of duplicates, quasi-replicas, or near-duplicates entries in their repositories. Because of that, there has been a huge effort from private and government organizations in developing effective methods for removing replicas from large data repositories. This is due to the fact that cleaned, replica-free repositories not only allow the retrieval of higher-quality information but also lead to a more concise data representation and to potential savings in computational time and resources to process this data.

Keywords: Genetic approach, Record Duplication, Database Integration.

1. INTRODUCTION

Genetics and Molecular Biology are keys for the understanding the mechanisms of many of the human diseases that have strong harmful effects. The rapid expansion of biomedical knowledge combined with the reduction in computing costs and spread of Internet access have created an enormous number of electronic data. This is especially true in biomedicine for the decentralized nature of the scientific community leading to a patchwork of diverse and heterogeneous databases' implementation and making access to and aggregation of data across databases very difficult. Databases are highly heterogeneous with respect to the data models they employ, the data schemas they specify, the query languages they support, and the terminologies they recognize. Heterogeneous database systems (HDBS) attempt to unify disparate databases by providing uniform conceptual schemas that resolve representational heterogeneities, and by providing querying capabilities that aggregate and integrate distributed data. Research in this area has applied a variety of database and knowledge-based techniques, including semantic data modeling, ontology definition, query translation, query optimization, and terminology mapping.

2. HETEROGENEOUS DATABASE INTEGRATION

The goal of a heterogeneous database system (HDBS) is to provide database transparency to users and application programmers. This means to provide a global and consistent database interface for applications as if the data were not distributed and all of the database management systems were of the same type. HDBS research emerges since the belief that heterogeneity at the level of constituent database systems will persist exists, despite standardization efforts. The process of heterogeneous database integration may be defined as the creation of a single, uniform query interface to data that are collected and stored in multiple, heterogeneous databases". Thus HDBSs are computational models and software implementations that grant heterogeneous database integration.

Different kind of heterogeneous database integration methods can be distinguished and are also useful in biomedicine.

- **Vertical Integration:** The aggregation of semantically similar data from multiple heterogeneous sources. For example, a centralized database to functional magnetic resonances collected in a country.
- **Horizontal Integration:** The composition of semantically complementary data from multiple heterogeneous sources. An example can be a system that provides complex queries across genetics and clinical information sources.
- **Integration for application portability:** The standardization of access to semantically similar information at disparate sources. For example, a universal database interface for decision-support applications that allows them to be shared across institutions with no modifications to their implementations.

HDBSs distinguish from distributed database systems (DDBS) even if they are frequently confused. The overlapping idea is the capability to provide a unified view and a common interface to data, physically stored in multiple locations. DDBS are more integrated and coordinated than HDBSs: DDBSs implement the same data model and query language and the core system uses the same distributed database management software. Moreover, in DDBSs, data fragmentation is designed to achieve efficiency and autonomy advantages of distributed computing. On the other hand, in HDBSs, the constituent

database existed prior to the establishment of the HDBS and their coordination is much weaker.
The characteristics of HDBSs can be summarized as follows:

- **Heterogeneity on data representations:** The core database in an HDBS can use different query languages and data models terminologies to represent the same real-world object and thus the same semantics. From this consideration follows that even if data are stored at multiple sites and could have identical semantics, the data representation and the data access methods at each site may be different.
- **Local autonomy:** HDBS grants rights to each constituent database on accessing, controlling and manipulating its own data independently from the central engine. Examples of such data manipulation or local database control can be changes of data representation or performance improvements.
- Bottom-up integration. A HDBS integrates data that were previously distributed, improving the interoperability aspects. On the other hand, a DDBS exploits the distribution of previously integrated data to obtain efficiency benefits. A bottom-up integration process requires that the HDBS provides interfaces to heterogeneous and preexisting information systems without needing of intensive local preexisting software modifications.

3. REQUIREMENTS

Providing a context to the challenges of heterogeneous database integration, we report a set of requirements:

- Database heterogeneity is a fact and a single model, for example for biological databases, is hard to achieve although the presence of standards.
- Heterogeneous databases systems must provide general query capabilities supported by efficiency procedures. These queries retrieve all data referred to a single object or a set of objects that satisfy the search criteria. Query procedures have not to depend from any particular application of information need.
- HDBS has to allow transparency at data level and thus users and applications are not required to know the existence, access methods, physical location or the schema of the underlying local databases.
- Permissions on local databases, for example writing access, are not required by common users and applications but they are restricted to local administrators.
- Since local database are designed and maintained to meet local needs, the underlying local database schemas can quickly change. Changes are made independently of the integrated database structure.
- As well as schemas, also the content of the local databases could change frequently by updating, deleting or inserting data.

One of the most important problems in heterogeneous database integration deals with query models: independently developed and maintained databases are heterogeneous with respect to their model of data storage and information retrieval. A query model must be known to database users or applications when queries are encapsulated into the executable commands.

A query model consists of four components:

1. The data representation abstract model, such as relational tables, flat files, and so on.
2. The schema of the represented data, for example in clinical context a schema of data can differ if starting from a set of patients and retrieving the set of relative physicians a system uses a single query or multiple queries.
3. The language for accessing data: syntax and semantics to query and retrieve data can be of high level (SQL for example) or low level (data fragmentation access).
4. The data format: it is common the use of abbreviations or codes for names or local formats, for example in the case of clinical phenotypes different systems may use different codes or abbreviations for the same object.

Heterogeneous database integration means also creating a single virtual query model that encapsulates the query models of underlying databases and allows users and programs to access data from the local databases using this virtual model. Other important issues in heterogeneous database integration are represented by the variety with which similar data are represented in different databases; this multitude of data schemas is called representational heterogeneity (RH). The most general type of heterogeneity is that of the data models themselves: aggregating data from relational, hierarchical, object oriented and flat file databases into a single representation is the first activity in schema integration. However, even if different database systems used the same model, for example a relational model, significant representational heterogeneity would remain such as structural differences, naming differences, semantic differences and content difference. In the following paragraphs these RH will be covered.

4. RELATED WORK

Record deduplication is a growing research topic in database and related fields such as digital libraries. Today, this problem arises mainly when data are collected from disparate sources using different information description styles and metadata standards. Other common place for replicas is found in data repositories created from OCR documents. These situations can lead to inconsistencies that may affect many systems such as those that depend on searching and mining tasks. To solve these inconsistencies it is necessary to design a deduplication function that combines the information available in the data repositories in order to identify whether a pair of record entries refers to

the same real-world entity. This problem was extensively discussed by Lawrence et al.. They propose a number of algorithms for matching citations from different sources based on edit distance, word matching, phrase matching, and subfield extraction. As more strategies for extracting disparate pieces of evidence become available, many works have proposed new distinct approaches to combine and use them.

Naive Bayes based on the original Fellegi-Sunter statistical model of record linkage, methods from Bayesian statistics such as Naive Bayes classifiers have been used to learn linkage rules. The main disadvantage of Naïve Bayes classifiers from a practical point of view is that they represent a black box system to the user. This means that the user cannot easily understand and improve the learned linkage rules. Elmagarmid et al. classify these approaches into the following two categories: 1) Ad-Hoc or Domain Knowledge Approaches—this category includes approaches that usually depend on specific domain knowledge or specific string distance metrics. Techniques that make use of declarative languages can be also classified in this category; 2) Training-based Approaches—This category includes all approaches that depend on some sort of training—supervised or semi-supervised—in order to identify the replicas. Probabilistic and machine learning approaches fall into this category. Newcombe et al. were the first ones to address the record deduplication problem as a Bayesian inference problem (a probabilistic problem) and proposed the first approach to automatically handle replicas. However, their approach was considered empirical since it lacks a more elaborated statistical ground. After Newcombe et al.'s work, Fellegi and Sunter proposed a more elaborated statistical approach to deal with the problem of combining evidence. Their method relies on the definition of two boundary values that are used to classify a pair of records as being replicas or not. Tools that implement this method, such as Febrl, usually work with two boundaries as follows: Positive identification boundary—if the similarity value lies above this boundary, the records are considered as replicas. Negative identification boundary—if the similarity value lies below this boundary, the records are considered as not being replicas. For the situation in which similarity values stand between the two boundaries, the records are classified as "possible matches" and, in this case, a human judgment is necessary.

5. GENETIC PROGRAMMING

Genetic programming is an extension of the genetic algorithm which has been first proposed by Cramer. Similar to a genetic algorithm, it starts with a randomly created population of individuals. Each individual is represented by a tree which is a potential solution to the given problem. From that starting point the algorithm iteratively transforms the population into a population with better individuals by applying a number of genetic operators. These operations are applied to individuals who have been selected based on a fitness measure which determines how close a specific individual is to the desired solution. The three genetic operators typically used in genetic programming are:

Reproduction: An individual is copied without modification.

Crossover: Two selected individuals are recombined into a new individual.

Mutation: A random modification is applied to the selected individual.

The algorithm stops as soon as either the configured maximum number of iterations or a user-defined stop condition is reached.

6. GENETIC PROGRAMMING APPROACH

An important problem in Linked Data is the discovery of links between entities which identify the same real world object. These links are often generated based on manually written linkage rules which specify the condition which must be fulfilled for two entities in order to be interlinked. We represent a linkage rule as a tree which is built from 4 basic operators:

Property: Creates a set of values to be used for comparison by retrieving all values of a specific property of the entity.

Transformation: Transforms the input values according to a specific data transformation function.

Comparison: Evaluates the similarity between the values of two input operators according to a specific distance measure. A user-specified threshold specifies the maximum distance. If the underlying properties do not provide any values for a specific entity, no similarity value is returned.

Aggregation: Aggregates the similarity values from multiple operators into a single value according to a specific aggregation function. Aggregation functions such as the weighted average may take the weight of the operators into account. If an operator is marked as required, the aggregation will only yield a value if the operator itself provides a similarity value.

The resulting linkage rule forms a tree where the terminals are given by the properties and the nodes are represented by transformations, comparisons and aggregations. The linkage rule tree is strongly typed i.e. it does not allow arbitrary combinations of its four basic operators.

7. CONCLUSION

Identifying and handling replicas is important to guarantee the quality of the information made available by data intensive systems such as digital libraries and e-commerce brokers. These systems rely on consistent data to offer high-quality services, and may be affected by the existence of duplicates, quasi replicas, or near-duplicate entries in their repositories. Thus, for this reason, there have been significant investments from private and government organizations for developing methods for removing replicas from large data repositories.

REFERENCES :

1. M. Bilenko and R. Mooney. Adaptive duplicate detection using learnable string similarity measures. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 39-48. ACM, 2003.
2. M. Bilenko and R. J. Mooney. Learning to combine trained distance metrics for duplicate detection in databases. Technical report, 2002.
3. T. Blickle and L. Thiele. Genetic Programming and Redundancy. 1994.
4. M. Carvalho, A. Laender, M. Goncalves, and A. da Silva. Replica identification using genetic programming. In Proceedings of the 2008 ACM symposium on Applied computing, pages 1801-1806. ACM, 2008.
5. C. Cortes and V. Vapnik. Support-vector networks. Machine Learning, 20:273-297,1995. 10.1007/BF00994018.
6. N. Cramer. A representation for the adaptive generation of simple sequential programs. In Proceedings of the First International Conference on Genetic Algorithms,volume 183, page 187, 1985.
7. M. G. de Carvalho, M. A. Gon_calves, A. H. F. Laender, and A. S. da Silva. Learning to deduplicate. In Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries, JCDL '06, pages 41-50, New York, NY, USA, 2006. ACM.
8. M. G. de Carvalho, A. H. F. Laender, M. A. Goncalves, and A. S. da Silva. A genetic programming approach to record deduplication. IEEE Transactions on Knowledge and Data Engineering, 99(PrePrints), 2010.
9. K. A. De Jong. An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, Ann Arbor, MI, USA, 1975.
10. M. Elfeky, V. Verykios, and A. Elmagarmid. Tailor: A record linkage toolbox. In Data Engineering, 2002. Proceedings. 18th International Conference on, pages 17-28. IEEE, 2002.