Data Integration: An Alternative Perspective

Tok Wang Ling   Mong Li Lee

Department of Computer Science
School of Computing
National University of Singapore
{lingtw, leeml}@comp.nus.edu.sg

Abstract

Given the rapid growth of the Internet and other on-line information repositories, it is increasing important to integrate a wider variety of data formats and data found on the Web. In this paper, we offer an alternative perspective on the integration of data from multiple sources such as traditional databases and semi-structured data sources on the Web. We observe that a lot of work on the integration of heterogenous data sources assume that the data is correct and complete. However, "dirty" data files are prevalent as a result of incorrect or missing data values due to data entry errors or typographical errors, inconsistent value naming conventions due to different field entry format and use of abbreviations, data value conflicts due to domain mismatch and incomplete information due to unavailability of data. Hence, we may have multiple records refering to the same real world entity. This not only contributes to the problem of handling ever-increasing amount of data, but also leads to inconsistent or duplicate information which is obviously undesirable. We will show that the process of detecting and removing "dirty" and duplicate records is not only necessary within a database, but it is also important when we integrate multiple databases.

1 Introduction

In the early nineties, a constantly increasing number of applications need to work on data which are scattered over independent file systems and/or legacy database systems of various types (hierarchical, navigational, relational etc). This has led to a lot of research in the field of *Federated Database Systems* (see [BK92, SG90] for an overview and classification of the various approaches. A federated database system consists of component database systems that are autonomous and yet participate in a federation to allow partial and controlled sharing of data. This is a compromise between no integration (in which users must explicitly interface with multiple autonomous databases) and total integration (in which the autonomy of each component database system is sacrificed so that users can access data through a single global interface but cannot directly access a database management system as a local user). Heterogeneity in the different database systems result from differences in data models and differences at the system level. Different data models lead to differences in structural representation of concepts, differences in constraints and differences in query languages. Different systems lead to differences in transaction management primitives and techniques (including concurrency control, commit protocols and recovery), hardware, system software and communication capabilities.

In federated database systems, the data originates from traditional databases and has a rigid and regular structure (or schema). The schemas of the component databases may be modeled using different data models. These schemas will be mapped to the equivalent schemas modeled using a canonical data model such as the object-oriented model [B91, KDN90] or Entity-Relationship model [NA87, LL95a]. This step facilitates subsequent data integration which typically involves schema integration and instance integration [KS91]. In schema integration, one needs to resolve various conflicts such as naming conflict, structural conflict, key conflict and constraint conflicts [LL95b, LL97]. In instance integration, one needs to decide whether two instances from different databases refer to the same entity in the real world and how to reconcile partial or inconsistent data values from the two matching instances [BDW91, IJ84, L95, L96]. [LFG+97] classify conflicts presented in data value into two categories: context dependent and context independent. Context dependent conflicts, also known as domain mismatch [K91] occurs because of the different interpretations inherent in different systems and applications, while context independent conflicts occurs because of errors.

The growth of the Internet and other on-line information repositories has greatly simplified the access to numerous sources of information. However, this growth has complicated tasks involving finding, extracting, merging or synthesizing such information. For example, a tourist planning a trip to New York cannot simply obtain from the Web a map showing all Italian restaurants close to New York World Trade Center. This is despite the fact that such information may well be available on the Web if maps of New York, lists of restaurants, and lists of tourist sites are combined. This is a very common scenario involving the extraction of data from the diverse information repositories across the Internet, and integrating the data from heterogeneous sources. Therefore, the challenge we face today is not information distribution, but information extraction and integration. While integration seems to be an old topic, the need to integrate a wider variety of data formats and data found on the Web has brought this topic to the forefront of research again [ACHK93, GMPQ+94, U97, C98]. This task is further complicated by the abundance of semistructured data on the Web. Semistructured data has no fixed schema, and the structure of the data is irregular or incomplete. Sometimes, the structure exists but has to be extracted from the data. Sometimes, the structure is ignored for browsing purposes. Examples of semistructured data include HTML documents whose links can be interpreted as edges in a graph, thus imposing some structure on the documents; and a bibliography file where some structure is imposed by fields such as author and title to an otherwise unstructured text file. With the recent emergence of XML (eXtensible Markup Language), research in the integration of semistructured data is expected to be even more active.

In this paper, we offer an alternative perspective on the integration of data from multiple sources such as traditional databases and semi-structured data sources on the Web. We observe that a lot of work on the integration of heterogenous data sources assume that the data is correct and complete. In practice, "dirty" data files are prevalent as a result of incorrect or missing data values due to data entry errors or typographical errors, inconsistent value naming conventions due to different field entry format and use of abbreviations, inconsistent data values due to domain mismatch and incomplete information due to unavailability of data. Hence, we may have multiple records refering to the same real world entity. This not only contributes to the problem of handling ever-increasing amount of data, but also leads to inconsistent or inaccurate information which is obviously undesirable. We will show that the process of detecting and removing "dirty" and duplicate records is not only necessary within a database, but it is also important for the query results obtained from the integration of multiple databases. A motivating example is given in Section 2. Section 3 presents a framework for data integration which takes into account complications that arise due to "dirty" and duplicate data, incomplete and inconsistent data. We conclude with directions for further research in Section 4.

2 Motivating Example

Most university professors maintain a Bibtex database to keep references. A published paper may be represented by several entries in a Bibtex database for various reasons: (1) incorrect or missing data values because of data entry errors or typing mistakes, (2) inconsistent value naming conventions because of different entry formats and the use of abbreviations such as 'page' vs 'pp', (3) incomplete information because certain data such as publisher or page numbers is not captured or available,

As a result, we encounter situations where several records may refer to the same real world entity while not being syntactically equivalent which complicates the integration of data. We can treat a set of records that refer to the same entity in the following ways. We can view one of the records as correct and the rest of the records as duplicates containing erroneous information. Then the objective is to cleanse the database of the duplicate records [ME97, SSU95, HS95]. In addition, we can also view each matching record as a partial source of information. Then these duplicate records should be merged to obtain one record with more complete information.

In the same way, when two or more professors work together on a paper, an immediate problem is to merge multiple Bibtex databases into one. Although all Bibtex databases have similar structure, the values in these databases may be partial or even inconsistent (conflicting). A common example is the authors of a paper. One professor may like to list all the authors in full names, but others may just indicate the first one or two authors. The order of the first name (which may be initial) and the last name of an author may also be different. Again, there may be missing or incomplete infor-

Record	Author	Title	Proceeding	Page No.	Year
1	Lee Alain Pier	A Language for XML	WWW Conference		1998
2	A.P. Lee	A Lang. for XML	Proc. of WWW Conferance	121-140	1898

Table 1: Example of two duplicate records.

Record	Author	Title	Proceeding	Page No.	Year
1	Lee, Alain Pier	A Language for XML	Proceedings of WWW Conference	121-140	1998

Table 2: Merged record contains more correct and complete information.

mation for the same reference, such as page number, publisher etc. Having cleansed each Bibtex database of "dirty" and duplicate records, the next step is to merge these databases and integrate them so that the information is complete. In the process, we may detect more duplicate records, and need to resolve additional inconsistent information. In fact, the same data cleansing process can be applied to the merged database as part of data integration.

In order to remove duplicated records from a dataset or multiple datasets, the main consideration is how to decide that two records are duplicate? This involves comparing two records to determine their degree of similarity. which also implies that the corresponding fields in the two records has to be compared. The comparison of fields to determine whether or not two syntactic values are alternative representations of the same semantic entity is also known as the **field matching problem** [ME96].

Table 1 shows two records, Record 1 and Record 2. At first glance, all the field values in both records look different. On closer examination, we note that the *Year* in Record 1 and Record 2 are very similar except for one digit difference. We also observe that "Lee" is common in the *Name* field of Record 1 and Record 2. Furthermore, "A.P." and "Lang." in the *Author* and *Title* fields of Record 2 could be an abbreviation of "Alain Pier" and "Language" in Record 1 respectively. We find that the *Proceedings* of Record 1 and Record 2 are actually the same except for the typographical error "Conferance" in Record 2 and the words "Proc. of" are missing in Record 1. Although Record 1 contains incomplete information (page numbers are missing), but the *Author, Title, Proceedings and Year Published* field values of Record 1 and Record 2 are very similar to each other. We may conclude that they are most likely to be duplicates and refer to the same published paper in the real world. The differences in the *Author* and *Proceedings* field values in Record 1 and Record 2 are typical of **different field entry format** problem. We can merge these two records to obtain one record with more complete information as shown in Table 2. Note that in the merged record, abbreviations have been removed, typographical errors corrected, and inconsistent naming conventions and data values resolved.

The process of determining the degree of similarity between two records is more efficient if we know the "key" of the records or the field(s) which are representative of the records. For example, it is obvious that *Author* and *Title* together are able to distinguish two records than *Proceedings* or *Year*. The above example illustrates how missing data values due to data entry errors or typographical errors, inconsistent value naming conventions due to different field entry format and use of abbreviations, data value conflicts due to domain mismatch and incomplete information due to unavailability of data can easily complicate the data integration process. This example is not only characteristic of the records found within one database, but it is also common of the records obtained by merging several databases.

3 Data Integration Framework

Figure 1 shows the architecture of a typical data integration system. The various data sources may be either structured files, WWW form interface, relational database or object-oriented databases. These sources can be viewed as containers of sets of tuples or records. Traditionally, translators are used to transform tuples from the databases such as relational database, object-oriented databases etc, to a form that can be further processed by the data integration system. With the growing number of information sources on the Web, specialized programs called wrappers which are specific to every web site are written to translate the data in the web site to a form that can be processed by the integration system. An integrated or mediated view of the data is then produced which allows the users or application programs to pose queries. To evaluate a query, the data integration system must also translate the query on the mediated schema into a query on the data sources.

One of the hardest problems in answering queries over a multitude of sources is deciding that two objects mentioned in two different sources refer to the same entity in the real world because each source employs its own naming conventions and abbreviations. This problem is exaberated by the prevalence of duplicate objects *within* the same source. A quick scan of the results returned by any search engine on the Web reveals plenty of duplicate information which adds to thousands and millions of tuples matching the query. Therefore, we propose that any attempt to integrate data from multiple sources should include a data cleansing phase to remove duplicate data first from within a database, and a cleansing phase to remove duplicate results returned by the integrator of multiple databases.

Figure 2 shows an alternative framework for data integration which incorporates cleansing of "dirty" and duplicate data. When building a data integration system, we can take either a warehousing or a virtual approach [HZ96]. In the warehousing approach, data from multiple sources is loaded into a warehouse, and all queries are applied to the warehoused data. While this requires that the warehouse be updated when the source data changes, the advantage is that adequate performance can be guaranteed at query time. In the virtual approach, data remains in the various sources. Queries to the data integration system are decomposed or reformulated at run time into the equivalent queries on the data sources. While this requires more sophisticated query optimization, the advantage is that data is guaranteed fresh at query time, particularly when the source data is changing frequently. For both approaches, our framework requires that each data source be first cleansed of "dirty" and duplicate data before it participates in any federation of information sources. This is done once and each cleansed data source can be easily updated when the original data source is updated.

The integrator/mediator needs to handle the following tasks which includes de-duplication:

1. Resolve domain mismatch. Domain mismatch arises when when we have conflict between the domains of equivalent attributes. For example, the value set for an attribute ExamScore may be in grades (A, B, C etc) in one database and in marks in another database. [LL97] distinguish two types of domain mismatch, namely reversible and irreversible, and present an algorithm to resolve them. [BFG+97] propose a Context Interchange Framework to support information exchange data among heterogenous and autonomous systems. A context mediator detects and reconciles semantic conflicts that arise by comparing the contexts associated with the various data sources.

2. Object identification. The removal of duplicate records from each data source will greatly facilitate the identification of matching records from multiple sources which represents the same entity in the real world. This identification process requires the use of keys or distinguishing fields in the records.

3. Merge and de-duplicate. The partial information in the matching records are merged to obtain one record with more complete information.

The results of any query on the integrated view is also cleansed of duplicates before it is given to the user or application program. If the warehousing approach is used in the data integration system, then we can apply the cleansing process to the warehouse of integrated data when it is first built. In this case, the query results can be directly returned. Note that translators/wrappers are required to transform the query results to a form that the application program or WWW based user interface can accept. In the same way, the translators/wrappers are needed to transform queries issued by the application program or WWW based user interface to a form that can be processed by the integrator/mediator.

In general, the duplicates elimination problem is difficult to handle both in scale and accuracy. We note that most systems deal with the problem by using domain specific heuristics such as the Smith-Waterman algorithm for comparing DNA and protein sequences [SW81] and variant entries in a lexicon [JR94]. In the WHIRL system, [C98] match objects across sources by using techniques from Information Retrieval. The standard method of detecting *exact* duplicates in a database is to sort the database and then check if the neighbouring records are identical [BD93]. In order to detect approximate duplicates in a database, the most reliable way is to compare every record with every other record in the

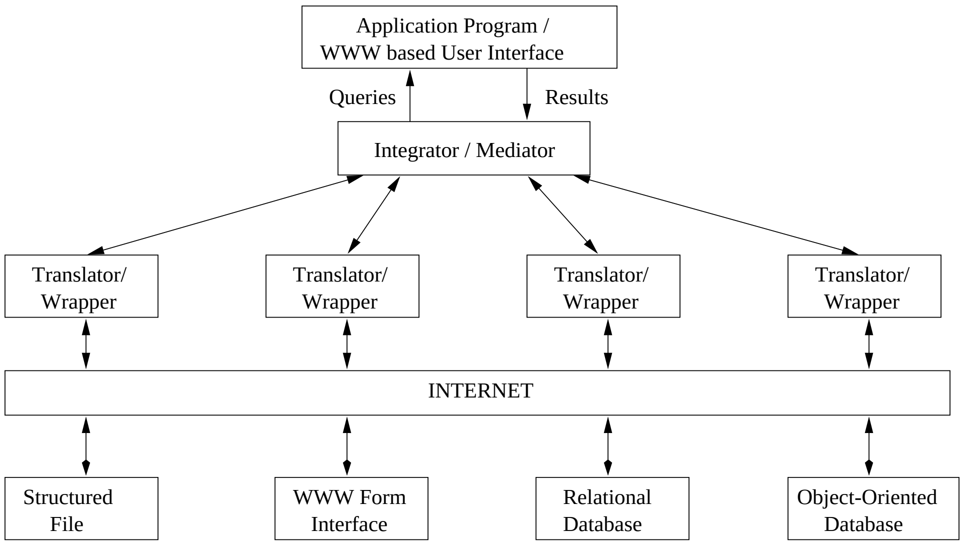

Figure 1: Architecture of a Data Integration System

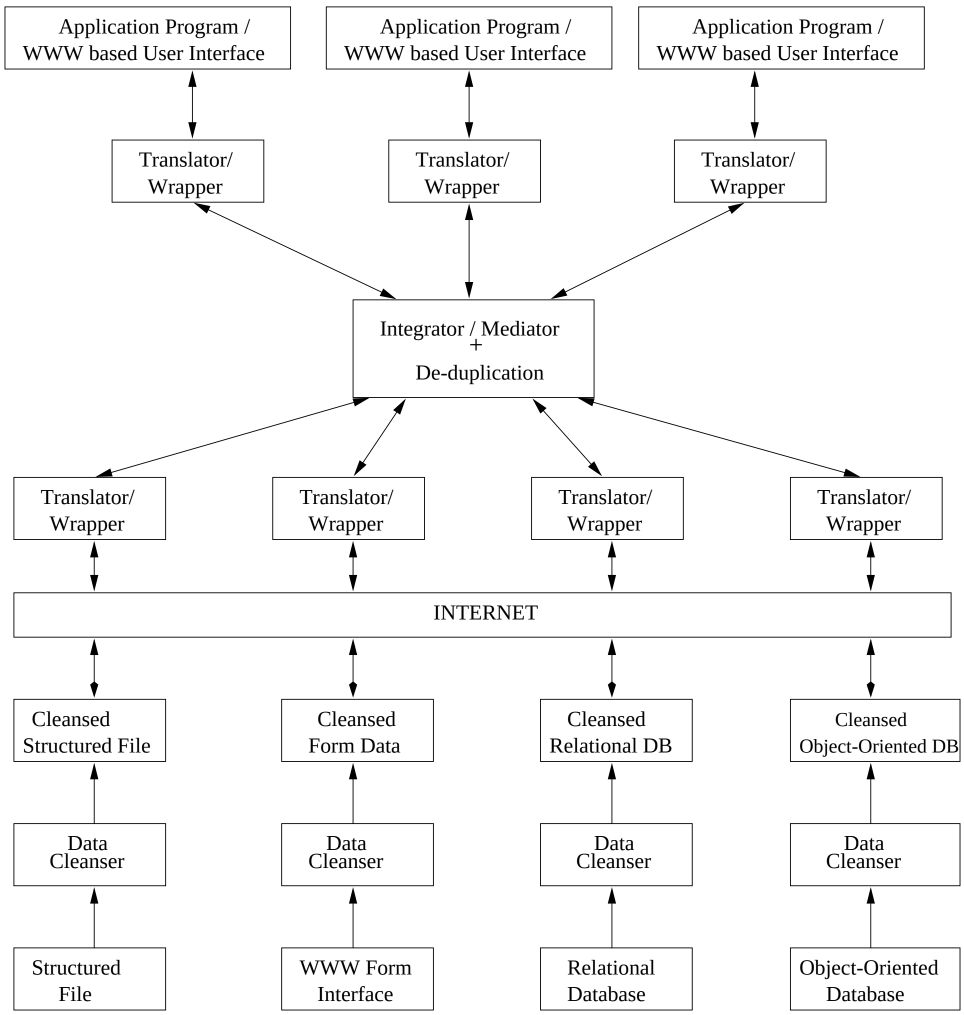

Figure 2: An Alternative Framework for a Data Integration System

database. However, this is a very slow process which requires N(N-1)/2 record comparisions, where N is the number of records in the database.

Our approach is to develop efficient and scalable techniques which cuts down the exhaustive comparisons of records [LLLK98]. This entails the preprocessing of records before sorting them so that potentially matching records will be brought to a close neighbourhood subsequently. The techniques developed include scrubbing data fields using external source files to remove typographical errors and abbreviations, tokenizing data fields and then sorting the tokens in the data fields. A window of fixed size is then moved through the sorted records to limit the comparisons for matching records. Each matching record is treated as a partial source of information and are merged to obtain one record with more complete information. Interested readers may refer to [LLLK98] for details of our cleansing techniques.

4 Future Works

In this paper, we have presented an alternative view on the integration of data from multiple sources. We have shown that the process of detecting and removing "dirty" and duplicate records is not only necessary within a database, but it is equally important in the the integration of multiple databases. This is because "dirty" data files are prevalent as a result of incorrect or missing data values due to data entry errors or typographical errors, inconsistent value naming conventions due to different field entry format and use of abbreviations, data value conflict due to domain mismatch and incomplete information due to unavailability of data. As a result, multiple records may refer to the same real world entity. The number of multiple records refering to the same real world entity grows rapidly when the number of sources to be integrated increases. We have implemented a data cleansing system which is able to detect and remove more duplicate records than existing methods. This cleansing system will be incorporated in a data integration system to cut down the duplicate results returned the integrator for any global query.

For future research, we want to investigate the problem of integrating incomplete information in semi-structured data, where although the data may have some structure, this structure is not as rigid, regular or complete as that required by traditional databases.

References

[ACHK93] Y. Arens, C.Y. Chee, C.N. Hsu and C.A. Knoblock. Retrieving and integrating data from multiple information sources. *Journal of Intelligient and Cooperative Information Systems*, 2(2), pp 127-158, 1993.

[B91] E. Bertino. Integration of heterogenous data repositories by using object-oriented views. *Proc. 1st Int. Workshop on Interoperability in Multidatabase Systems*, pp 22-29, 1991.

[BD93] D. Bitton and D.J. DeWitt. Duplicate record elimination in large data files. *ACM Transactions on Database Systems*, 1995.

[BDW91] O.P. Buneman, S.B. Davidson and A. Watters. A Semantics for Complex Objects and Approximate Answers. *Journal of Computer and System Sciences*, 43(1), pp 170-218, 1991.

[BFG+97] S. Bressan, K. Fynn, C.H. Goh, M. Jakobisiak, K. Hussien, H. Kon, T. Lee, S.E. Madnick, T. Pena, J. Qu, A. Shum and M. Siegel. The COntext INterchange mediator prototype. *Proc. ACM SIGMOD/PODS Joint Conference*, 1997.

[BK92] M.W. Bright and S.H. Pakzad. A taxonomy and current issues in multidatabase systems. *IEEE Computer, 25(3)*, pp 50-60, 1992.

[C98] W. Cohen. Integration of heterogenous databases without common domains using queries based on textual similarity. *Proc. ACM SIGMOD*, 1998.

[GMPQ+94] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman and J. Widom. The TSIMMIS Project: Integration of Heterogenous Information Sources. *Proc. IPSJ Conference*, pp 7-18, 1994.

[HS95] M. Hernandez and S. Stolfo. The merge/purge problem for large databases. *Proc. of ACM SIGMOD Int. Conference on Management of Data* pages 127-138, 1995.

[HZ96] R. Hull and G. Zhou. A framework for supporting data integration using the materialized and virtual approaches. *Proc. ACM SIGMOD*, pp 51-61, 1996.

[IJ84] T. Imielinski and W.L. Jr. Incomplete Information in Relational Databases. *Journal of ACM*, 31(4), pp 761-791, 1984.

[JR94] C. Jacquemin and J. Royaute. Retrieving terms and their variants in a lexicalized unification-based framework. *Proc. of the ACM-SIGIR Conference on Research and Development in Information Retrieval* pages 132-141, 1994.

[KDN90] M. Kaul, K. Drosten and E. Neuhold. ViewSystem: Integrating Heterogenous Information Bases by Object-Oriented Views. *IEEE 6th Int. Conference on Data Engineering*, 1990.

[K91] W. Kent. Solving domain mismatch and schema mismatch problems with an object-oriented database programming language. *Proc. 17th Int. Conference on Very Large Databases*, 1991.

[KS91] W. Kim and J. Seo. Classifying schematic and data heterogeniety in multidatabase systems. *IEEE Computer*, 24(12), pp 12-18, 1991.

[L95] L. Libkin. Normalizing Incomplete Databases. *Proc. of ACM PODS*, pp 219-230, 1995.

[L96] A.Y. Levy. Obtaining complete answers from incomplete databases. *Proc. 22nd International Conference on Very Large Databases*, pp 402-412, 1996.

[LL95a] T.W. Ling and M.L. Lee. Relational to Entity-Relationship Schema Translation Using Semantic and Inclusion Dependencies. *Journal of Integrated Computer-Aided Engineering*, John-Wiley Publishers, 2(2), pp 125-145, 1995.

[LL95b] M.L. Lee and T.W. Ling. Resolving Structural Conflicts in the Integration of Entity-Relationship Schemas. *Proc. of 14th International Conference on Object-Oriented and Entity-Relationship Modelling (OOER95)*, 1995.

[LL97] M.L. Lee and T.W. Ling. Resolving Constraint Conflicts in the Integration of Entity-Relationship Schemas. *Proc. of 16th International Conference on Conceptual Modeling*, 1997.

[LLLK98] M.L. Lee, H. Lu, T.W. Ling, Y.T. Ko. Cleansing Data for Mining and Warehousing. *submited for publication*, 1998.

[LFG+97] H. Lu, W. Fan, C.H. Goh, S.E. Madnick and D.W. Cheung. DIRECT: Discovering and Reconciling Conflicts for Data Integration. *Proc. of the 7th IFIP 2.6 Working Conference on Data Semantics (DS-7)*, 1997.

[ME96] A.E. Monge and C.P. Elkan. The field matching problem: Algorithms and applications. *Proc. of the 2nd Int. Conference on Knowledge Discovery and Data Mining* pages 267-270, 1996.

[ME97] A.E. Monge and C.P. Elkan. An Efficient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Records. *SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1997.

[NA87] S.B. Navathe and A.M. Awong. Abstracting Relational and Hierarchic Data with a Semantic Data Model. *Proc. 7th Int. Conference on Entity-Relationship Approach*, pp 277-306, 1987.

[SG90] A.P. Sheth and S.K. Gala. Federated database systems for managing distributed, heterogenous and autonomous databases. *ACM Computing Surveys, 22(3)*, pp 183-236, 1990.

[SSU95] A. Siberschatz, M. Stonebraker, and J.D. Ullman. Database research: achievements and opportunities into the 21st century. A report of an NSF workshop on the future of database research *SIGMOD RECORD*, March 1996.

[SW81] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology* 147:195-197, 1981.

[U97] J.D. Ullman. Information Integration using Logical Views. *Proc. of International Conference on Database Theory (ICDT)*, 1997.