8-16-1996

An Information Discovery Process for Interoperable Heterogeneous Databases

A. Zisman
Department of Computing, Imperial College, az@doc.ic.ac.uk

J. Kramer
Department of Computing, Imperial College, jk@doc.ic.ac.uk

An Information Discovery Process for Interoperable Heterogeneous Databases

A. Zisman* J. Kramer**
az@doc.ic.ac.uk jk@doc.ic.ac.uk
Department Of Computing - Imperial College
180 Queen's Gate - London, UK-SW7 2BZ
Phone: +44-171-5948239 Fax: +44-171-5818024

1. Introduction

An important issue when dealing with interoperability of heterogeneous databases is *information discovery*. This is the location and identification of information that is relevant, identical, similar or related to the requested data of a query. As outlined by Sheth [7], in the future, the primary issue will not be to efficiently process the data that is known to be relevant, but to determine which data is relevant and where it is located.

Some approaches have been proposed in the literature addressing the information discovery problem for a large number of databases. One approach [1, 8] uses an external indexing scheme. However, the autonomy of the involved databases is not strictly respected, since they have to provide both the content and the structure to other databases. It is not specified how the external indexes are updated when components are added and removed from the system. For simplicity, it is assumed that all the data is public, ignoring privacy aspects. Another approach [5, 6] to inter-node relationship discovery is based on the use ofa high level "context abstraction" that defines objects using *Global Concepts* (GCs). The databases are related to the GCs by *link weights*, based on areas of interest. It is not specified how to organise the GCs, how to access the data of a database, and how to add and remove databases to/from the system. A framework and system called FINDIT [2, 3] was proposed to dynamically educate the users about the available information space. FINDIT is atwo-level approach based on *coalition* and *service*. The databases are organised upon information interest and optionally in geographicalproximity. Each participating database contains an object-oriented *co-database* storing information about coalition and services in which it is involved. However, to guarantee protection of data the system uses checking mechanisms to identify if a data can be accessed by a specific user. It is difficult to handle queries involving more than one type of information.

In this extended abstract we propose an algorithm to perform information discovery for a large number of autonomous heterogeneous databases. The challenge is to support the discovery process after evolving from the situation where single databases are accessed and manipulated in isolation, to the case where these databases support interoperation, permitting applications to also access remote information. The approach aims to avoid the use of centralised structures (dictionaries and repositories), containing information about the databases of the system, and integrated schemas, in the traditional sense (global schemas). It also tries to avoid broadcasting to theentire network. The algorithm is designed to reduce the number of databases searched during the discovery process, is executed in parallel and recursively, avoiding cycles, and tries to preserve the privacy of the shared data.

2. An Architecture

An architecture is proposed to assist in the distributed information discovery process. A detailed description of the architecture can be found in [9, 10].

In the proposed architecture the databases are arranged into *federations* as illustrated in figure 1. A federation consists of a set of databases willing to share data with each other. The criteria to form a federation are based on the shared data of the databases and the databases that are allowed to access this shared data. Inside a federation, the shared data of a database is public for all the other participants in this federation. There is no communication between two federations. However, one database can participate in different federations (overlapping), sharing a distinct set of data in each of these federations. The idea of using federations is to provide privacy of the shared data and avoid the need for *negotiation* [1, 2, 4, 6], in order to access identified data between the requester of the data and the owner (access rights).

Inside a federation the databases are organised into *groups*. The idea of groups is to make the universe of search smaller, facilitating the information discovery process. A group of databases is formed based on the type of data shared by these components. For instance, a federation can contain a *university group*, a *hospital group* and an *institute group* (federation 1 in figure 1). It is possible to have a database that participates in more than one group in the same federation.

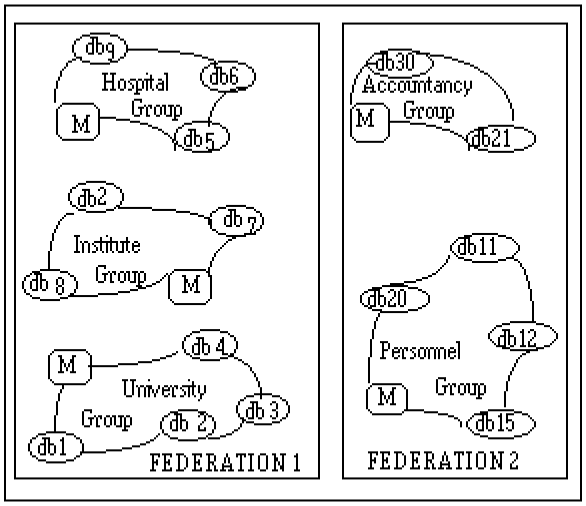

Figure 1: Federations of heterogeneous databases.

Each group contains a special component named *master* (M). The master records the location of the different databases participating in a federation. It contains the more up to date information about the related group, since the addition (removal) of a database is first notified in the master of the related group. During a discovery process, whenever a requested data is not found in any of the databases of a group, the master of this group is consulted, to identify a database not yet visited which possibly contains this data.

Apart from the master, each database contains an additional structure named *locate* (LOC), for each federation that it participates in. Actually, the master and the LOC contain the same "type" of information, structured in the same manner. They are composed by the names of the groups of the related federation and *specialised terms,* forming a natural hierarchy of names (from general to specific terms). Associated with these terms there are references to the concerned databases. The proposed structure need not to be balanced. It depends on the level of specialisation achieved for each group of the federation. A balanced configuration limits the natural hierarchy of the terms. The terms are organised in the hierarchy depending on the applications of the system.

A database contains a set of other additional structures for each federation that it participates in. These structures are the data that it shares with the other databases of the federation (public schema), expressed both in its native data model and in a canonical data model, to solve the heterogeneous aspects. There is also descriptions of semantics information of the public data. These semantics descriptions are related to the differences (conflicts) involving the data, such as: naming (synonyms and homonyms), structural and representative (type mismatch, format, units), and different levels of abstractions.

Each federation contains a human *manager* to help to decide about the different groups of the federation, and the names and terms associated to these groups. During the construction of the LOC structures, the manager is responsible to identify the differences in terminologies of the involved databases, and to specify general terms to homogenize these differences. Alternatively, each database has an *administrator* (DBA) that decides about different data to be shared, federations that it participates in, and so on.

The proposed architecture permits dynamic evolution of the system, allowing a database (or a group of databases) to join or leave a federation. During the evolution process, the master structures of the involved groups are updated first. The next step consists in updating the LOC structures of the databases inside the related federation.

3. The Process

The proposed method performs the discovery process in a distributed manner, without using centralised structures (repositories and dictionaries) or broadcasting to all databases. The idea is to consult a reduced number of databases when searching for information. For requested data, the process is performed inside a single federation to preserveprivacy. In this federation, a group of databases related to the required data is identified to be visited. Thereby, the method avoids the situation where a database containing the required data is specified, but it is not allowed to share this data with the requester. During the process, a visited database has either the requested data or knowledge about other databases that may have this data. The approach is executed in parallel and recursively, and avoids cycles (for each requested information a database is visited at most once).

Suppose the situation where a user of a database *DB'* (source component) wants to access information *d'* that does not exist init. The first step consists in identifying which is the federation that *DB'* participates that is related to the required data. After this, the LOC structure of *DB'* associated to this federation is traversed. A specific group of databases having the sets of shared data related to *d'* is identified. These databases are called *possible target components*. The identification is executed based on the terms used to classify the databases in the hierarchy and the keywords associated to the query of *d'*. The next step consists in confirming if any of these possible target components contains *d'*, i.e. the query for *d'* is sent in parallel to these possible target components. If a possible target component does not contains *d'*, then its LOC structure is consulted, in order to identify other "new" possible target components. The process is recursive and it is guaranteed to terminate either when *d'* is found (success), or *d'* is not found (failure). When a visited database does not contain *d'* and the search in its LOC structure does not identify any other related database not yet visited, the master of the related group is visited. Since the master contains the most up to date information, it may identify a "new" possible target component. In this case, the request is sent to it. Notice that the discovery process can assist in the update of the LOC structures of the databases involved in this process. That is, when the LOC structure of a database *dbj* (identified as a possible target component after traversing the LOC structure of database *dbi*) contains more up to date information, then the LOC of *dbi* is updated.

The process terminates whenever: (a) the correct data is found (success), (b) all the databases in the related group are visited and none of them contains the data (failure), (c) the user asks to suspend the process. In cases (a) and (c) the process has to be cancelled, since it is executed in parallel. We suggest sending *cancellation messages* starting at the source component and traversing the same path as the requests. To assist in this, the approach uses a direct graph built during the discovery process. In the graph (figure 2) the nodes are the source component, the visited databases and the master (when visited). An edge *eij = dbi dbj* of the graph, denotes that *dbj* was visited after traversing the LOC structure of *dbi.* Thus, the cancellation messages can traverse the same graph (forwards). In case (b), the source component has to be notified about the non existence of the data. The idea is to traverse the graph backwards. Whenever a database does not have the requested data and does not know about any other "new" possible target component, it notifies its parent node in the graph. When this parent is notified by all its sons, it notifies its own parent, and so on, until achieving the source component (cf. diffusing algorithms).

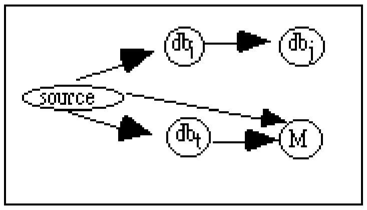

Figure 2: An example of the graph.

The avoidance of cycle is performed by making each database record the fact that it has been queried for particular data *d'*. We guarantee that the process terminates since, for data *d'*, the search is executed inside a group which contains a finite number of databases. However, the assurance that existing searched data is found depends on the way in which the databases are grouped and the hierarchical structure organised (associated terms).

4. Conclusion

We presented an approach that performs a distributed information discovery process in a large structured organisation of databases. It aims to avoid the use of integrated schemas and centralised structures. The method is assisted by the use of hierarchical structures. It attempts to reduce the number of databases searched and to preserve the privacy of the shared data. The main idea is to visit a database that either contains the requested data or knows about another database that possible contains this data. The correctness of the process is related to the way that the groups are formed inside each federation and associated terms are chosen. It is necessary to gain experience to evaluate the approach through modelling. We plan to implement a prototype to validate the proposed ideas.

References

[1] R. Alonso, D. Barbara, and S. Cohn. Data sharing in a large heterogeneous environment. In *7th International Conference on Data Engineering*, pages 305-313, Held Kobe, April 1991. IEEE.

[2] A. Bouguettaya. *A Dynamic Framework for Interoperability in Large Multidatabases.* PhD thesis, faculty of Graduate School of the University of Colorado, 1992.

[3] A. Bouguettaya, S. Milliner, and R. King. Resource location in large scale heterogeneous and autonomous databases. *Journal of Intelligent Information System*, 5(2), 1995.

[4] D. Heimbigner and D. McLeod. A federated architecture for information management. *ACM Transaction on Office Information Systems,* 3(3):253-278, July 1985.

[5] S. Milliner and M. Papazoglou. A scalable architecture for interactions between autonomous distributed database nodes. Submitted for publication.

[6] S. Milliner and M. Papazoglou. Reassessing the roles of negotiation and contracting for interoperable databases. *Int'l Workshop on Advances in Databases and Information Systems, Russian ACM SIGMOD,* May 1994.

[7] A.P. Sheth. Semantic issues in multidatabase systems. *SIGMOD RECORD,* 20(4):5-9, December 1991.

[8] P. Simpson and R. Alonso. A model for information exchange among autonomous databases. Technical Report, Princeton University, May 1989.

[9] A. Zisman. Towards interoperability in heterogeneous database systems. Technical Report 11, Department of Computing, Imperial College of Science, Technology and Medicine, 1995.

[10] A. Zisman and J. Kramer. An architecture to support interoperability of autonomous database systems. To appear in *2nd International Baltic Workshop on DB and IS*, Tallin-Estonia, 12-14 June 1996.

---

* This work has been supported by Brazilian foundation CNPQ, grant no. 200209/94-9.

** This work has been partially funded by Information System Interoperability project, grant no. ECAUS003.