Mediators in the Architecture of Grid Information Systems*

Peter Brezany[1], A. Min Tjoa[2], Helmut Wanek[1], Alexander Wöhrer[1]

[1] Institute for Software Science
University of Vienna, Liechtensteinstrasse 22, A-1090 Vienna, Austria
{brezany|woehrer}@par.univie.ac.at, wanek@chello.at
[2] Institute for Software Technology and Mutimedia Systems
Vienna University of Technology, Vienna, Austria
tjoa@ifs.tuwien.ac.at

Abstract. Across a wide variety of fields, huge datasets are being collected and accumulated at a dramatical pace. The datasets addressed by individual applications are very often heterogeneous and geographically distributed. In this paper, we describe our extensions and improvements to the reference implementation of the OGSA-DAI Grid Data Service prototype, an infrastructure that allows remote access to Grid databases, in order to provide a Virtual Data Source – a clean abstraction of heterogeneous/distributed data for users and applications. By picturing general applicable access scenarios, we are showing the great need for such a Grid data mediation service as well as the compliance with important requirements of virtual data sources.

1 Introduction

Grid computing can be defined as applying resources from many computers in a network – at the same time – to a single problem; usually a problem that requires a large number of processing cycles or access to large amounts of data. At its core, it enables devices – regardless of their operating characteristics – to be virtually shared, managed and accessed across an enterprise, industry or workgroup. This virtualization of resources places all of the necessary access, data and processing power at the fingertips of those who need to rapidly solve complex business problems, conduct compute-intensive research and data analysis, and engage in real-time.

The World Wide Web began as a technology for scientific collaboration and was later adopted for e-business. Scientists foresee - and indeed, we are experiencing - a similar trajectory for Grid technologies [9]. Many research funding organizations (e.g. e-Science Programme in U.K. [6]) and commercial companies (e.g. IBM [23]) are driving the benefits of Grid computing beyond its academic and research roots into business enterprises. This includes the introduction of the Grid technology into key industries – such as aerospace, automotive, financial markets, government and life sciences. All these applications demand an infrastructure and tools for data management and analysis. This implies interfaces for federating databases [24] and techniques for metadata generation and management alongside other data issues.

A wide variety of major e-Science applications [13, 19, 21] are supported by the Globus Toolkit [26], a community-based, open architecture, open source set of services and software libraries. The Open Grid Services Architecture (OGSA) Framework, the Globus-IBM vision for the convergence of Web Services and Grid computing, takes the ideas of Web Services and extends them to provide support for Grid Services.

The development of OGSA technical specification is ongoing within the Global Grid Forum[7] inside the tasks called the *Open Grid Services Infrastructure (OGSI)*. The Globus project is developing the *Globus Toolkit 3.0 (GT3)*, which is based on OGSI mechanisms; the first experimental implementation, *GT3 Alpha Grid Services*, is already available.

As mentioned before, Grid computing began with an emphasis on compute-intensive tasks, which benefit from massive parallelism for their computational needs, but are not data intensive; the data that they operate on does not scale in portion to the computation they perform. In recent years, this focus has shifted to more data-intensive applications, where significant processing is done on very large amounts of data.

* The work described in this paper is being carried out as part of the research projects "Modern Data Analysis on Computational Grids" and "Aurora" supported by the Austrian Research Foundation.

According to the analysis reported in [20], there is a dearth of Grid applications that use databases to store scientific data - almost all existing applications use files. A significant research and development effort has already been devoted to efficient management, placement and replication of Grid files [25]. However, if the Grid is to support a wider range of applications, both scientific and commercial, then database integration into the Grid will become important. Therefore, within the context of OGSA activities, the Global Grid Forum Database Access and Integration Services (DAIS) Group developed a specification [1] for a collection of OGSI-compliant Grid database services. The first implementation [2] of the service interfaces, *OGSA-DAIS Release 2*, is already available.

The next logical step is the support for federating data resources, as depicted in fig. 1, which is vital to the success of the Grid. The alternative of forcing each application to interface directly to a set of databases and resolve federation problems internally would lead to application complexity, and duplication of effort. Database federation has been comprehensively studied [24], and implementations are available from vendors. However, these are unlikely to meet the needs of all Grid applications. The factors that make Grid database federation different include for example high dynamic flexibility, extreme performance, and semantic aspects [3].

The central component of fig. 1 is the *Mediator*, which is realized as a special Grid Service (GDMS) exposing a relational database and an XML database to various applications as one cohesive data repository. By means of the *SDE* and *perform* ports [8] the applications can query the mediation service features (metadata), status information and the database data, respectively.

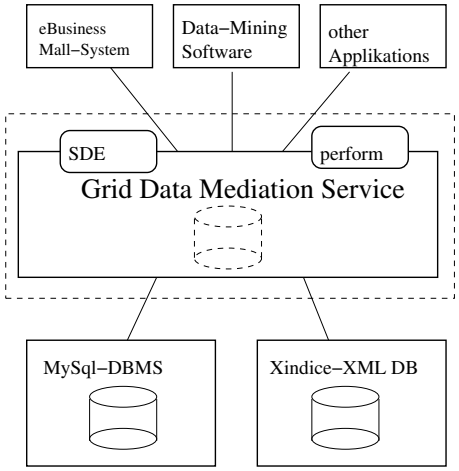


Fig. 1. Grid Data Mediation Service (GDMS) providing a *virtual data source* (VDS) for different applications, handling and hiding the heterogeneity of the two involved databases

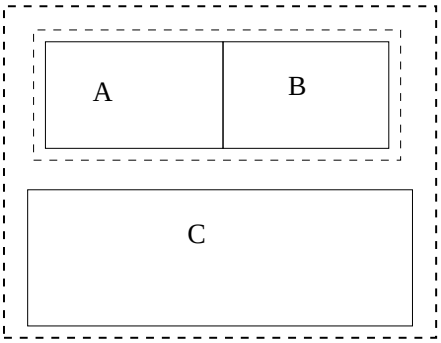
This paper describes the design and implementation of the first (to our best knowledge) mediation system for databases integrated into the Grid. Our contribution significantly leverages the functionality of the OGSA-DAI reference Grid Data Service implementation.

The remaining part of the paper is organized as follows. In section 2 we are delineating why there is a great need for data mediation on the Grid and what important requirements have to be fulfilled by the mediators. The kernel part of the paper is section 3, which introduces the OGSA-DAI reference implementation and our extensions and improvements to this architecture. Section 4 discusses related work whereas section 5 briefly outline the future work. The paper is closed with our conclusions in section 6.

2 Data Access Scenarios

Our example access scenario is derived from the domain of health care. Let's suppose you want to mine data sources from different sites as depicted in fig. 2. The data of the two involved hospitals is distributed over the

three departments A,B and C. Although the two hospitals store the same information about their patients, the data structures are different. Let's assume that the name information of hospital one is represented by the patient's full name, and the name information of an patient at hospital two is divided into first name (fn) and last name (ln). Hospital one has *no central* patient database and so the data is divided and stored in two different databases - the administrative info in the administration department database (in fig. 2 called site A) and the medical information in the care unit database (in fig. 2 called site B). The other informations provided like blood type, date of first treatment, day of birth and adress have the same structure but are accessible over different names. To *materialize* the virtual data source (i.e. to reconstruct it from its fragments), the following operations are required: R = (A JOIN B) UNION C.



A(pid,name,adress,dob)
B(pid,bloodtype, first_contact)
data provided by hospital one

C(pid, ln, fn, adr, birthday, bt, fc)
data provided by hospital two

Fig. 2. Vertical and horizontal fragmentation of the virtual data source over the three departments A, B, C at 2 hospitals in the left part and the attributes of the relations in the right part

Next, consider a number of e-business companies who wish to build up one central information point for their costumers. Therefore, the new mall application needs access to a number of (legacy) data base systems that manage or use the information about the product prices in the participating companies of the mall system. Each of the systems was developed independently to meet the needs of different applications and may contain the required information in a different structure.

```
<ROWSET>
 <ROW>
  <pid>AS4990</pid>
  <name>Adam Smith</name>
  <adress>2730 Rock Road, 12</adress>
  <dob>24/07/1980</dob>
  <bloodtype>A</bloodtype>
  <first_contact>01/02/1990</first_contact>
 </ROW>
 ...
</ROWSET>
```

(a) Result of the join between the data of department A and B

```
<ROWSET>
 <ROW>
  <pid>TD4990</pid>
  <fn>Tom</fn>
  <ln>Duncan</ln>
  <adr>2730 South East Road, 12</adr>
  <birthday>12/03/1985</birthday>
  <bt>A+</bt>
  <fc>03/07/1997</fc>
 </ROW>
 ...
</ROWSET>
```

(b) Result of department C

Fig. 3. Partial query results

It is clear, that *virtualization* almost always involves a loss of data access performance. Since many applications may use the Grid primarily for high performance, the mediator is discardable. Virtualized

access can be provided, but is not the only behavior. An application that wants high performance is able to directly access the underlying sources by requesting a GDS for every data source, e.g., in order to apply optimizations specific to a particular data format.

Each data resouce understands the Grid Data Mediation Service queries and returns the results in XML format as depicted in fig. 3, which shows the query results of the operations and the fact that the results are similar but the structures are different. Fig. 4 shows the final result for the query 'SELECT * FROM PATIENT' after the mediation process has been applied.

```
<ROWSET>
 <ROW>
  <p_id>TD4990</p_id>
  <p_name>Tom Duncan</p_name>
  <p_adr>2730 South East Road, 12</p_adr>
  <p_dob>12/03/1985</p_dob>
  <p_bt>A+</p_bt>
  <p_fc>03/07/1997</p_fc>
 </ROW>
 <ROW>
  <p_id>AS4990</p_id>
  <p_name>Adam Smith</p_name>
  <p_adress>2730 Rock Road, 12</p_adress>
  <p_dob>24/07/1980</p_dob>
  <p_bt>A</p_bt>
  <p_fc>01/02/1990</p_fc>
 </ROW>
 ...
</ROWSET>
```

Fig. 4. The final result of the virtual data source after the mediation

With our architecture described in section 3 we meet the concerns of the following important requirements of virtual data sources [3]:

1. When more than one data resource is specified, the Grid must provide the ability to link them together, even if they have different data structures, to produce a single logical target that gives consistent results.
2. When linking data resources, the Grid must provide the ability to use data in one resource as the matching criteria or conditions for retrieving data from another resource, i.e. perform a sub-query. As an example, it should be possible to compare predicted gene sequences in a local database against those defined in a centralized curated repository.
3. The Grid must be able to construct distributed queries when the target data resources are located at different sites, and must be able to support heterogeneous and federated queries when some data resources are accessed through different query languages. The integrated access potentially needs to support retrieval of data that match common search criteria and matching conditions. This process may involve temporary storage being allocated for the duration of the retrieval.

3 Architecture of the Grid Data Mediation System (GDMS)

The GDMS architecture is illustrated in fig. 5. The left side shows the simplified functional structure of the OGSA-DAI release 2 prototype. As one can easily see, it provides three Grid Services: a GDSR (Grid Data Service Registry), GDSF (Grid Data Service Factory) and a GDS (Grid Data Service).

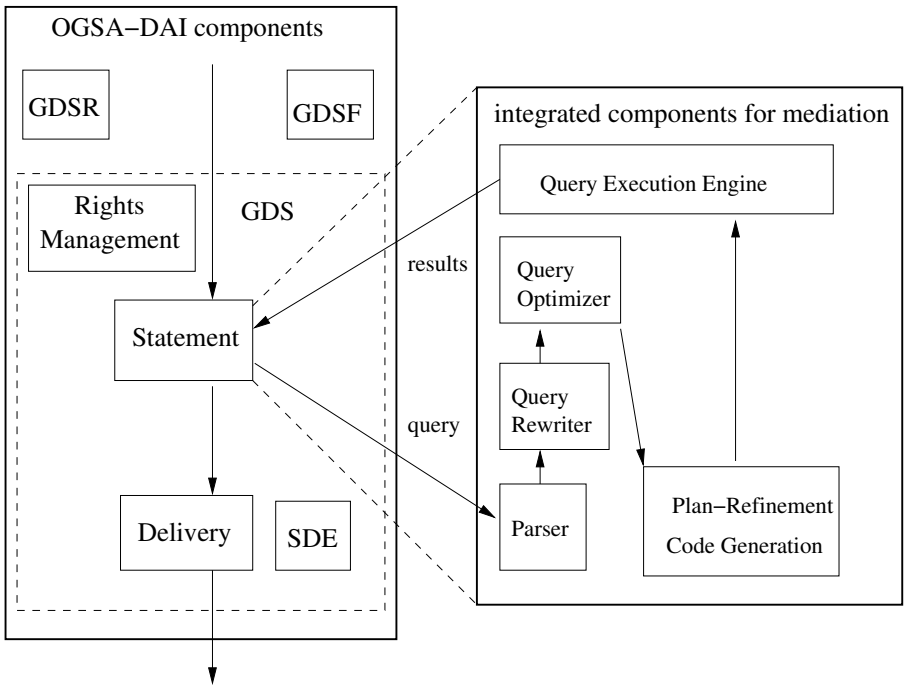


Fig. 5. In the left part of the figure, a simplified structure of the current OGSA-DAI architecture is described, in the right the integrated GDMS modules replacing the standard statment handling

The *GDSR* provides a directory facility for OGSA-DAI services. OGSA-DAI services can be registered with a GDSR, providing their GSH[3] and information describing the service, allowing them to advertise their capabilities (metadata about a service) to the world. Clients can then search the GDSR for services that match their requirements and the GDSR will return the GSHs of these services. For example, a client can search a GDSR for information on GDSFs that manage particular data resources of interest to the client.

The *GDSF* provides a service creation facility, creating GDSs which facilitate access to a particular data resource. GDSFs advertise information on the GDSs they can create using the GDSR. A client can contact a GDSF and request creation of a GDS serving a particular data resource. GDSFs are persistent services, they are created when the Grid Services container[4] that hosts them is created and they are destroyed when this container is destroyed. A GDSF configuration file, accessed when a GDSF is created, can specify the GDSRs that the GDSF should register with on creation, thereby making clients aware of its existence. More importantly, this configuration file specifies the database management system and database or collection, i.e. the data resource, with which the GDSs created by the GDSF will interact. This file contains some of the *metadata* needed to interact with the data resource like the physical and logical schema of the data resource or technical aspects as the supported query language. Also the *mapping schema* (for example, the schema in fig. 6 needed for our scenario in section 2) for describing the mediation task and characteristics of the VDS can be stored there for very complex or often used mediation tasks. For highly dynamic federations, this metadata can also be passed to the GDSF at runtime.

The *GDS* is the primary OGSA-DAI service. GDSs provide access to data resources using a document-oriented model: a client submits a data retrieval or update request in the form of an XML document called GDS-Perform documents – which allows a detailed delivery description, which are instructions for the Delivery activity in fig. 5 to specify how the results (e.g. via output stream or GridFtp) or status (e.g. completed or detailed error messages) of the operation should be returned to the client or a third party,

[3] Grid Service Handle, is a globally unique name that distinguishes a specific Grid service instance from all other Grid service instances

[4] in our case this is the Globus 3 Framework hosted within the Jakarta Tomcat web server

to be included. The GDS executes the request and returns an XML document holding the results of the request. Unlike GDSFs and GDSRs, GDSs are transient services. When created by a GDSF they are given a finite lifetime. When that lifetime ends, the GDS is destroyed. But clients can extend the lifetime of a GDS, if required. When a GDS is created, it is configured by a GDSF using the already mentioned GDSF configuration file which specifies the data resource the GDS will interact with and the data retrieval/update operations the GDS can perform.

```
<MappingSchema>
 <Table name="patient">
   <union>
     <join>
      <source key="pid" id="A">
        <map dest="p_id" src="pid"/>
        <map dest="p_name" src="name"/>
        <map dest="p_adr" src="adress"/>
        <map dest="p_dob" src="dob"/>
      </source>
      <source key="pid" id="B>
        <map dest="p_bt" src="bloodtype"/>
        <map dest="p_fc" src="first_contact"/>
        <map dest="p_id" src="pid"/>
      </source>
     </join>
     <source key="pid" id="C">
      <map dest="p_id" src="pid"/>
      <map dest="p_name" src="concatenate(fn,ln)"/>
      <map dest="p_adr" src="adr"/>
      <map dest="p_dob" src="birthday"/>
      <map dest="p_bt" src="bt"/>
      <map dest="p_fc" src="fc"/>
     </source>
   </union>
 </Table>
 ...
</MappingSchema>
```

Fig. 6. Mapping schema for the example scenario given in fig. 2

The *Rights Management* is supporting the concept of *Virtual Organizations* (VOs)[5] where you can define access rights to the data sources individually for each VO.

With the help of *SDEs* (Service Data Element)[6] of a GDS, the metadata for the service instance is provided. This includes the scheme, physical as well as logical, information of the data resource the GDS connects to and the activities[7] the GDS instance supports.

For the integration of the results from the different data sources, our GDMS uses a *mapping schema*, which defines mapping information between the virtual data source and the participating data sources. The XML instance outlined in fig. 6 is an example for a mapping schema in order to mediate the data resources given in our example scenario in section 2.

[5] groups of individuals, institutions and resource providers which decide to share resources across the different organisations to constitute and behave as a single VO. The members of the VO may be distributed in space and may use heterogeneous platforms

[6] information about Grid service instances, which is structured as a set of named and typed XML elements, encapsulated in a standard container format

[7] e.g. read only or full access, supported delivery mechanisms

The main element *MappingSchema* consists of a subelement *Table* which represents the mapping information for a virtual table contained in the *VDS*. This subelement contains other subelements such as *union* and *join*. Union operations are defined by enumerating the fragments to merge in the *union* element, which are in our example the results of a join operation and the results of a projection. The *join* element contains the information required for the join operation. The participating data sources are kept in the *source* elements with their key names.

On the right side of fig. 5, the applied *textbook architecture*, we are using for our mediator, is illustrated. This architecture was used, for example, in the IBM's Starburst project [22] and can be used for any kind of database system including centralized, distributed, or parallel systems [16]. Below we briefly describe the main components of this architecture.

Parser. In the first phase, the query is parsed and translated into an internal representation (e.g. a query graph) that can be easily processed by the later phases.

Query rewrite. It transforms a query in order to carry out optimizations that are good regardless of the physical state of the system. Typical transformations are the elimination of redundant predicates or simplification of expressions. In our system, where data is distributed, this part also selects the partitions of a table that must be considered to answer a query. With the help of the *mapping schema* given in section 3 and the extended *logical schemas* [11] stored (for each data source) in the GDSF, the basic part of the mediation is done.

Query Optimizer. This component carries out optimizations that depend on the physical state of the system. E.g. the optimizer is able to choose another *replica* (if specified in the GDSF) when one replicated data resource isn't responding in order to materialize the VDS anyhow.

Plan Refinement/Code Generation. This module transforms the plan produced by the optimizer into an executable plan.

Query Execution Engine. This element provides generic implementations for every operator. Our query execution engine is based on an iterator model [10]. In such a model, operators are implemented as iterators and all iterators have the same interface. As a result, any two iterators can be plugged together (as specified by the consumer producer relationship of a plan), and thus, any plan can be executed. Another advantage of the iterator model is that it supports the pipelining of results from one operator to another in order to achieve good performance.

4 Related Work

Basically the work presented here addresses integration of database sources into the Grid and database mediation. Grid database access is being developed and researched by the Database Access and Integration Services Working Group (DAIS-WG) [1, 20] in close cooperation with the OGSA-DAI project [2].

Database mediation has been studied for quite a while [27, 15] and especially for distributed and heterogeneous systems [5]. Recently there is also a trend to use XML either to transport queries and/or query results or to describe database schemas and even to store the data itself [4, 17, 18].

Optimization of query execution for parallel and distributed databases [16] is also an extremely relevant topic. Since Grid Services are distributed over the Internet which mostly offers very indeterministic connections (in terms of speed of data transfer or reliability) adaptive optimization strategies that can immediately react to changes in the environment seem promising [14].

The GDMS presented here is also intented to build a basis for future Grid data mining projects. A survey of parallel and distributed data mining can be found in [28]. And finally semantic models and ontologies could be used to automatically generate mappings between schemas of different and heterogeneous databases [12].

5 Future Work

A prototype implementation of the GDMS for horizontal partitioning has been developed to prove the feasibility of the described concept. To increase the usability and performance of our mediation system our future works include the following research items:

– Extend the parser/query rewriter in order to allow more complex and colorful user queries.
– Refining the *mapping schema* and with it the functionality of the system to support mediation tasks as e.g. *outer join* and *union all* elements in the schema
– Include more sophisticated methods for *replica* selection.
– To evolve the developed concepts so that they can be applied in a *parallel mediation service* to increase the performance.

6 Conclusions

In this paper, we have described the research effort, which focuses on the application and extension of the Grid technology to the mediation of datasources. The integration of two comprehensive research fields like mediation on the one hand and the younger and still evolving Grid technologies on the other hand is as ambitious as extensive. In this research effort, we have defined an extendable and flexible system architecture which is able to reduce complexity when accessing heterogenous and distributed data. It provides a clean abstraction of heterogeneous data for users/applications, supports a wide variety of data formats and is useable with a subset of the well known standard query language SQL92. By providing different views of data, pre-defined or determined at run-time, according to various applications needs, it fulfills requirements for high dynamic and flexible data access.

References

1. Global Grid Forum Database Access and Integration Services Working Group. http://www.cs.man.ac.uk/grid-db/.
2. Open Grid Services Architecture Data Access and Integration (OGSA-DAI). http://www.ogsadai.org.
3. Malcolm P. Atkinson, Vijay Dialani, Leanne Guy, Inderpal Narang, Norman W. Paton, Dave Pearson, Tony Storey, and Paul Watson. *Grid Database Access and Integration: Requirements and Functionalities*, Feb 2003. Global Grid Forum 7.
4. C. Baru et al. XML-based information mediation with MIX. SIGMOD '99, 1999.
5. P. Brezany, M. Bubak, M. Malewski, and K. Zajac. Large-scale scientific irregular computing on clusters and grids. Proceedings of the 2nd International Conference on Computational Science, ICCS 2002, Amsterdam, April 21-24, 2002, April 2002.
6. National e Science Centre. http://umbriel.dcs.gla.ac.uk/NeSC/general/.
7. Global Grid Forum. http://www.globalgridforum.org.
8. I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration, July 2002.
9. I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. Intl. J. Supercomputer Applications, 15(3), 2001.
10. Goetz Graefe. Query evaluation techniques for large databases. *ACM Computing Surveys (CSUR)*, 25(2):73–169, 1993.
11. Neil Hong, Amy Krause, Susan Malaika, Gavin McCance, Simon Laws, James Magowan, Norman W. Paton, and Greg Riccardi. *Grid Database Service Specification*, Feb 2003. Global Grid Forum 7.
12. Richard Hull. Managing semantic heterogeneity in databases: a theoretical prospective, 1997.
13. NASA's Information Power Grid (IPG). http://www.ipg.nasa.gov/.
14. Zachary G. Ives, Daniela Florescu, Marc Friedman, Alon Levy, and Daniel S. Weld. An adaptive query execution system for data integration, 1999.
15. Vanja Josifovski and Tore Risch. Comparison of amos ii with other data integration projects (working paper), 1999.
16. Donald Kossmann. The state of the art in distributed query processing. *ACM Computing Surveys (CSUR)*, 32(4):422–469, 2000.
17. Kangchan Lee, Jaehong Min, and Kishik Park. A Design and Implementation of XML-based mediation Framework(XMF) for Integration of Internet Information Resources. In *Proceedings of the 35th Hawaii International Conference on System Sciences - 2002*, 2002.
18. Hui Lin, Tore Risch, and Timour Katchaounov. Object-oriented mediator queries to xml data.
19. Grid Physics Network. http://www.griphyn.org.
20. Norman W. Paton, Vijay Dialani, Tony Storey, Malcom P. Atkinson, Dave Pearson, and Paul Watson. *Database Access and Integration Services on the Grid*, Feb 2002.
21. EU DataGrid Project. http://eu-datagrid.web.cern.ch/eu-datagrid/.

22. IBM Starburst Project. http://www.almaden.ibm.com/cs/starwinds/starburst.html, 1984-1992.
23. IBM Research. http://www.research.ibm.com.
24. Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3):183–236, 1990.
25. H. Stockinger. Database replication in world-wide distributed data grids. Phd Thesis, University of Vienna, November 2001.
26. Globus toolkit. http://www.globus.org/toolkit/.
27. G. Wiederhold. Mediators in the architecture of future information systems. The IEEE Computer Magazine, March 1992.
28. Mohammed J. Zaki. Parallel and distributed association mining: A survey. *IEEE Concurrency*, 7(4):14–25, /1999.