Peer-to-Peer Overlays and Data Integration in a Life Science Grid

Curt Cramer, Andrea Schafferhans, and Thomas Fuhrmann

Institut für Telematik, Universität Karlsruhe (TH), 76128 Karlsruhe, Germany
{cramer|fuhrmann}@tm.uka.de
Lion Bioscience AG, 69123 Heidelberg, Germany
andrea.schafferhans@lionbioscience.com

Abstract. Databases and Grid computing are a good match. With the service orientation of Grid computing, the complexity of maintaining and integrating databases can be kept away from the actual users. Data access and integration is performed via services, which also allow to employ an access control.

While it is our perception that many proposed Grid applications rely on a centralized and static infrastructure, Peer-to-Peer (P2P) technologies might help to dynamically scale and enhance Grid applications. The focus does not lie on publicly available P2P networks here, but on the self-organizing capabilities of P2P networks in general. A P2P overlay could, e.g., be used to improve the distribution of queries in a data Grid.

For studying the combination of these three technologies, Grid computing, databases, and P2P, in this paper, we use an existing application from the life sciences, drug target validation, as an example. In its current form, this system has several drawbacks. We believe that they can be alleviated by using a combination of the service-based architecture of Grid computing and P2P technologies for implementing the services.

The work presented in this paper is in progress. We mainly focus on the description of the current system state, its problems and the proposed new architecture. For a better understanding, we also outline the main topics related to the work presented here.

1 Introduction

There are many cases in which the integration of data from different sources gains additional value [Wat03]. This is why most of the proposed applications of Grid Computing have data integration at their core.

In the Grid community, it is a common assumption that certain sets of Grid participants form *virtual organizations* (VOs) that exist only for a limited time and aim at jointly achieving a certain, well-defined goal [FKT03]. The members of a VO, however, may be limited in their rights to access data from other members of their VO. Therefore, data access should be exposed to VO members only via services. The access control is then performed on the basis of these services.

A VO could, e.g., be formed by different life science research laboratories, both in universities and in companies. As continuous example in this paper, we use a typical application from the life sciences, namely validation of drug targets. Here, the involvement of a protein in a disease and the potential of treating the disease by activating or inhibiting the protein is evaluated with a variety of biochemical and in-silico methods.

The potential targets and the data gathered in validating them are the intellectual property of the respective pharmaceutical companies and their contracted biotech partners. These companies typically maintain huge database libraries containing information about expression of the proteins in different tissues, protein-protein interactions, interactions between proteins and small molecules (potential drugs), etc. Some of the relevant information, e.g. protein and gene sequences, is publicly available from different research institutions, but comes in a variety of formats, often flat files due to historical reasons.

Today, pharma companies replicate the contents of the publicly available databases to local repositories, where they are pre-processed and integrated with the in-house data to allow complex queries while ensuring that proprietary information does not leave the company. This means, however, that data not replicated in the in-house repository cannot be accessed. Furthermore, many pharma companies outsource biochemical analyses to specialised biotech companies (e.g. profiling the interactions of a target protein or solving 3D structures) and need to integrate their results with the in-house data. Finally, today's globally operating pharma companies are faced with the problem of integrating data from their various sites, with often separately evolved database systems.

Here, Grid Computing with its service orientation is a promising approach to reduce the database maintenance overhead of individual institutions. Ideally, the required external databases could be remotely queried in their up-to-date state without the need to repeatedly copy and pre-process them locally. E.g., database services could be defined with the OGSA-DAI toolkit [Mis04] and then be used to integrate the different data sources.

Collaboration of separate companies within a VO is very unlikely to result in open sharing of data between different parties. Especially biotech companies that perform services for different pharma companies need to give their individual clients access only to specific subsets of their results. Furthermore, since information gathered about a target protein is confidential, it should not be accessible to competitors. Pharma companies do not want to let their competitors know about the specific research performed by deducing patterns from the queries made by them. Hence some kind of "query obfuscation" needs to be employed to reduce this risk.

In order to protect the intellectual property of a company, all queries have to be executed on local database servers. To relieve the database servers from the high load posed on them, distributed query evaluation is needed. There are two extreme types of distributed evaluation. First, with *data shipping*, all data relevant to a query is transferred to one node which then has the burden to do all the processing associated with the query and afterwards return the result.

Query shipping, on the other hand, transfers (parts of) the query to the data sources, where intermediary results are generated. These intermediary results are exchanged between the data sources in order to construct the final result. Mixed forms of these two approaches are conceivable.

As mentioned above, shipping data to third parties might be problematic, since companies do not want their competitors to access their proprietary and sensitive data. Therefore, in this paper, we present our ongoing work to evaluate architectures for query shipping which allow to reduce the processing load put on the nodes. The approach we are currently following is to use a P2P overlay for scheduling queries on node subsets of VOs. The constitution of these subsets is controlled by particular companies, which is why the subsets can be considered trustworthy. The size of the subsets can be adapted dynamically to scale the system with increasing load, since the nodes in the subsets organize themselves in a P2P manner.

In order to provide interoperability with existing applications, our P2P query execution mechanisms on these subsets will be exported as a standard Grid service, e.g. using the interfaces defined by OGSA-DAI. This hides the functionality of query scheduling behind OGSA-DAI's standardized interfaces. By doing this, we can also employ the service description methods of OGSA-DAI to semantically integrate different data sources using appropriate ontologies.

The paper is structured as follows: In section 2, we describe the current architecture of the application example explained above. Then, we point out the major issues with it. After outlining the work related to ours in section 3, section 4 describes and discusses the architecture of an improved system which is to remove the previously mentioned issues. We then conclude in section 5 by highlighting the future directions of our work.

2 Problem Definition

In the last section, we gave a general overview of the drug target validation application. The currently employed architecture of this system is depicted in figure 1. Each pharma company replicates (some of) the content stored in public databases into its own database. Additionally, data generated by contracted third parties is merged into the local database. This data is only to be used by the respective pharma company. Note that although depicted as if the data were sent over the Internet, it is quite common to send these data on CDs to the pharma companies. Both the data retrieved from public databases and from third party contractors are manually integrated with the proprietary data of the pharma companies. Then, in-silico methods can be applied to the data.

This simple and pragmatic solution has several drawbacks:

1. Since the public databases have a size of several gigabytes, the data transfer takes a long time. Additionally, bandwidth and storage are aggessively consumed by the repeated replication of the full databases, even if only parts of them are actually accessed. Likewise, even databases that are only rarely

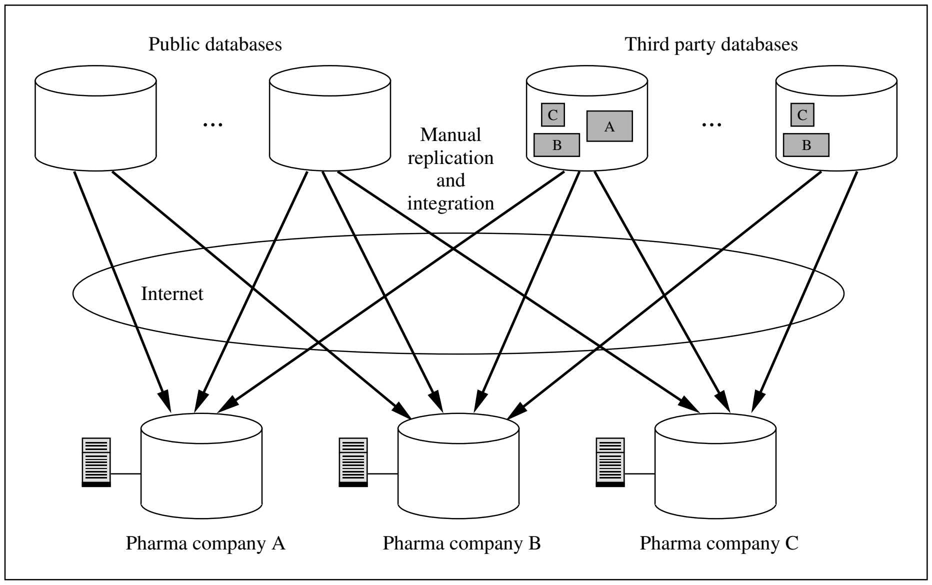

Fig. 1. Current System Architecture

accessed need to be replicated to the local system in order to be integrated with the other databases.

2. Content stored in the public databases changes frequently. Hence, even more bandwidth and processing power is consumed by constant updating. Furthermore, it sometimes occurs that the data is outdated once the in-silico calculations have finished. The full cycle then has to be repeated.

We aim at eliminating these drawbacks while retaining the strong confidentiality constraints posed on the system: No proprietary data is allowed to leave a company and no other party than the pharma company itself must know about the analyses performed on the data.

Note that these security requirements are implicitly fulfilled in the system depicted in figure 1: All computations are performed at the respective pharma companies. If a third party were to do these computations, it might deduce some knowledge about the projects run by the pharma companies. We do not want the third party contractors or the public database providers to be able to gain insight into the intellectual property of a pharma company. This includes relations between facts that could be derived from query patterns.

3 Related Work

The work presented in this paper is related to several research areas. In this section, we point out their main characteristics and especially those relevant to our work.

3.1 Grid Computing

The term *Grid Computing* is associated with the sharing of resources in virtual organizations (VOs) formed by multiple institutions [FKT03]. A VO is formed for following a common goal which is to be jointly achieved. As in real life, the participation of institutions is constrained by rules. Example applications of Grid computing include manufacturing, where each supplier in the chain is member of the VO, high energy physics with its vast storage requirements and many more. Grid applications typically have vast processing or storage requirements.

Grid computing employs a service-oriented approach and is heavily focused on standardization [FKNT03]. Services are defined and accessed by using standardized languages. Since all services use the same set of standards and languages, convenient service discovery through registries is enabled. Participants of a VO might be limited in their rights to access certain resources and therefore resource access has to be restricted using security services [FKNT03].

As discussed by Watson [Wat03], the combination of different data sources provides added value. Many applications rely on databases, however existent database management systems (DBMSs) are not especially designed for Grid environments. Since huge investments have been made for the development of DBMSs and for maintaining the data, one cannot simply develop new DBMSs for Grids. Existent DBMSs have to be integrated with the Grid [Wat03].

Kunszt et al. [KG03] discuss how databases can be fit into the standards and the Grid service model. An example implementation for wrapping commodity databases into Grid services has been produced in the OGSA-Database Access and Integration project (OGSA-DAI) [Wat03,Mis04]. This toolkit is easy to use and adapt to new requirements.

3.2 Peer-to-Peer Overlays

Peer-to-Peer (P2P) networks are powerful way to enable resource sharing between hosts, called peers here. The peers connect to each other using certain rules for creating a virtual network topology which is overlaid to the underlying physical network. Links in this topology are realized as transport channels while the P2P protocols are situated at the application layer in the layer model of the Internet.

Early P2P overlays were so-called *unstructured* P2P networks. There, the formation of the overlay topology is very flexible and does not prescribe the choice of overlay links or the number of links. The flexibility of unstructured P2P networks allows nodes to choose “good” links, e.g. with low delay. However, since the topology is random, data items cannot be deterministically located. This is the reason why unstructured P2P networks like Gnutella [Kan01] employ flooding or random walks in order to locate items.

To overcome the scalability and efficiency problems of flooding-based P2P protocols, structured overlay networks, e.g. Distributed Hash Tables (DHTs), have been proposed. There, the topology formation rules are quite rigid to assure routing efficiency. Each node is assigned a pseudo-random identifier, based on which packet routing is done in the overlay. Nodes have to choose their links according to a pre-defined scheme in order to provide routing guarantees. I.e., a node commonly maintains $O(\log n)$ links and thus the average lookup path length in the overlay also is $O(\log n)$, where n is the number of nodes participating in the overlay. A widespread example of structured overlays is Chord [SMK+01].

The virtualized topology of structured overlays commonly does not match with the topology of the underlying physical network. Therefore, latencies for sending messages in the overlay are a multiple of the latency observed when sending the message between the source and sink in the underlay. This multiple is called the *relative delay penalty (RDP)*. A lot of work has been done on reducing the RDP in structured overlays, however success is limited by the respective type of the overlay [GGG+03].

A lot of applications have been proposed to be run on P2P overlay networks. In unstructured overlays, file sharing was a typical application. Research on structured overlays also started with this application type, however there has been a shift to other applications like multicast recently. In this light, we see P2P overlays as a powerful technology to perform application-specific routing independent of some physical network constraints.

As was already mentioned by Foster [FKT03,FI03], current P2P systems have not yet reached the state where protocol integration and interoperability are desired. However, both Grid computing and P2P systems are concerned with the sharing of resources and have similar objectives [FI03]. Therefore, the combination of insights from both approaches seems to be promising.

3.3 Distributed Database Systems

In the described application, there is a multitude of databases which are to be *integrated* in order to run combined queries over them. Since these databases are individually controlled and have different schemas — they in fact can also use different data models — the system can be seen as a *distributed multidatabase system* [ÖV91].

There are several points to be handled in distributed multidatabase system (Distributed Multi-DBMS). If the Multi-DBMS should have its own global schema, the schemas of the individual databases have to be integrated. This is a very complex task. We do not consider schema integration here since we assume that the data coming from different databases are disjoint and queries mainly involve joining them. There, the attributes to join on are already specified in the queries.

Another point which is not of interest here is transaction management. We assume that the public and the third party databases cannot be modified by the companies using them. Therefore, no serialization problems can occur.

Apart from schema integration and transaction management, distributed query execution is another important aspect of Distributed Multi-DBMSs. As explained by Özsu et al. [ÖV91], a Distributed Multi-DBMS can be seen as a layer on top of the individual DBMSs. Each individual database runs an instance of this layer. This layer can be seen as a wrapper which exports the database's

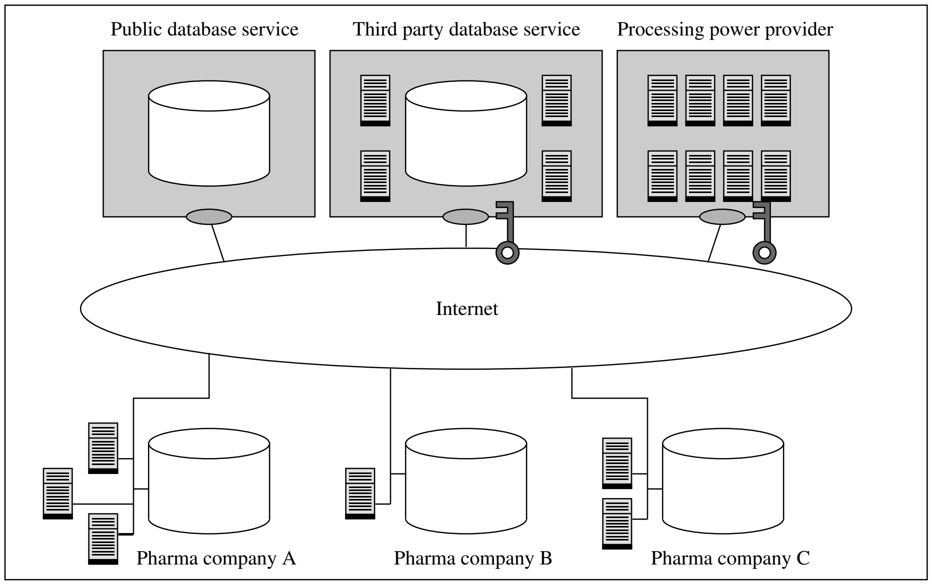

Fig. 2. Improved Service-Oriented Architecture

functionality to be used in a distributed system. It is responsible for distributing queries across multiple databases, which can be decomposed into several steps.

In a simplified form of the steps presented by Özsu et al. [ÖV91], the distributed execution of queries involves the splitting of a query across the databases, the translation of queries into the respective languages for each database and the integration of results.

The access to the databases in our example could e.g. be exported as an OGSA-DAI [Mis04] service. OGSA-DQP, an engine for distributed query processing [AMP$^+$03], can perform distributed queries over data sources defined with OGSA-DAI.

4 Architectural Considerations

We have outlined how the system is currently structured and employed in section 2. Along with that, we have pointed out the major problem with this system. A natural way to improve it, reducing the occuring problems, is to cast database access and integration, and processing into *services*. The architecture of this improved, Grid-like system is shown in figure 2.

In the remainder of this section, we discuss the advantages, disadvantages, and consequences of restructuring different parts of the system as services.

4.1 Public Databases

By offering access to the public databases as a Grid service, the need to repeatedly download, replicate, and integrate the data is cancelled. Database maintenance then only has to be performed by the database host. The consumption of bandwidth is reduced since with the service-based approach, only the required data is queried from the databases.

4.2 Third Party Contractors

The access to data provided by third party contractors can also be done via services. Since these data are not public and only to be accessed by pharma companies paying for it, AAA mechanisms (authentication, authorization, and accounting) are required to restrict access.

4.3 Public and Third Party Databases

The transformation of data access into a service solves problems, but also introduces new ones. In section 2, we have stated that we do not want outsiders to deduce knowledge about currently running projects from the request pattern of the pharma companies. With the current architecture, this problem does not occur since little knowledge is gained from the fact that a full database is used by a pharma company. Using the service-based approach, the data providers get to see the request patterns of the pharma companies, yielding more comprehensive knowledge.

In order to achieve the goal, the actual requests have to be disclosed from the data providers as far as possible. To this end, bogus requests might have to be interspersed in the stream of requests originating from a pharma company. This of course has to be done before the stream of requests reaches the data provider's database.

Another advantage of providing data access via a service is that new databases or databases other than the purchased ones can be automatically discovered. If the data services are semantically annotated, pharma companies can learn of databases which might be useful to them, but they have not purchased yet.

4.4 Processing Power Providers

The Grid approach also gives rise to new business opportunities. E.g., in figure 2, a provider of processing power was added to the system. Since data integration does not necessarily have to be performed at the respective pharma companies any more, data processing could be outsourced by them. By seeing these processing providers as contracted partners, we assume their machines to be trusted parties. Since potentially all pharma companies participating in the system might want to outsource their processing resources, again AAA mechanisms are required to restrict access to the nodes installed by the processing providers. For the strict security and confidentiality reasons mentioned above, processing jobs originating from different pharma companies must be completely shielded from each other. Neither must a pharma company A see what kind of processing is performed by company B, nor must it see any of its data involved in the processing.

4.5 Distributed Querying

In order to execute queries over the distributed databases, we plan to use the toolkits mentioned in section 3. By exporting database access as an OGSA-DAI service and using OGSA-DAP for distributed querying, we can use existing standards and implementations to get a working system soon. Then, we plan to transparently improve the distributed querying facility by employing a P2P overlay approach.

As shown [AMP$^+$03] and in a simplified version in figure 3, distributed querying requires *query planning*. A query plan is the result of compiling and optimizing a query. As indicated by the dashed lines, this plan is partitioned across the databases, which execute their parts of the query. The partial results have to be integrated afterwards, i.e. by the operation depicted by the node in the overlapping region of the partitions.

There are two degrees of freedom in this: First, the partitioning of the query plan. This might require redundant data storage at multiple sites. And second, the implementation of the physical operators for integrating the results (typcially join operations). PIER is an example of how join operations can be implemented using a DHT [HHL$^+$03]. Both degrees of freedom require the definition of a cost model for making appropriate choices.

E.g., if the databases are arranged in a P2P overlay, the link weights (delay, bandwidth) could be used to guide the selection of particular partitioning. The integration of the partial results does not necessarily have to take place at one of the database sites. We assume that each pharma company defines a subset of all nodes in the VO as trustworthy, i.e. by using AAA mechanisms. Then, the task of integrating partial results can be assigned to any node in this trustworthy node subset.

This approach of shipping queries together with partial results to nodes also has to take into account the transmission and processing costs of jobs. We envision a P2P overlay formed by the set of trustworthy nodes which organizes itself according to a cost model for query shipping. Queries can then be routed in this network towards suitable nodes, providing a load balancing mechanism. By dynamically adding more nodes to the query overlay, e.g. by renting more processing power from a provider, the system can be scaled in a self-organizing way.

5 Future Work

The next steps in our project involve a detailed analysis of the available database types and typical queries posed to the system. With this information, we can

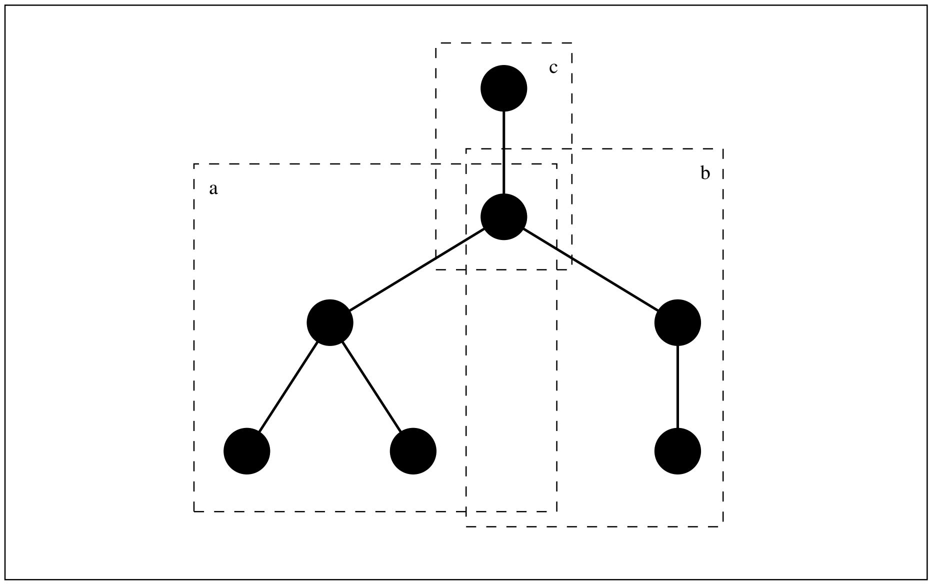

Fig. 3. Query Plan for Distribution

further specify the requirements of the distributed query processing component. Important topics there are the formation of the query overlay, the implementation of physical database operators using P2P approaches, and suitable AAA mechanisms and policies for achieving the rigid security requirements of the application.

References

[AMP$^+$03] M. Nedim Alpdemir, Arijit Mukherjee, Norman W. Paton, Paul Watson, Alvaro A. A. Fernandes, Anastasios Gounaris, and Jim Smith. Service-Based Distributed Querying on the Grid. In *Proceedings of the International Conference on Service-Oriented Computing*, 2003.

[BFH03] Fran Berman, Geoffrey Fox, and Tony Hey, editors. *Grid Computing: Making the Global Infrastructure a Reality.* John Wiley & Sons Ltd, Wiley Series in Communications Networking & Distributed Systems, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, 2003.

[FI03] Ian Foster and Adriana Iamnitchi. On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, Berkeley, CA, USA, February 2003.

[FKNT03] Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke. *The physiology of the Grid*, pages 217–249. In Berman et al. [BFH03], 2003.

[FKT03] Ian Foster, Carl Kesselman, and Steven Tuecke. *The anatomy of the Grid*, pages 171–197. In Berman et al. [BFH03], 2003.

[GGG$^+$03] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proceedings of the SIGCOMM 2003 conference*, pages 381–394. ACM Press, 2003.

[HHL$^+$03] Ryan Huebsch, Joseph M. Hellerstein, Nick Lanham, Boon Thau Loo, Scott Shenker, and Ion Stoica. Querying the Internet with PIER. In *Proceedings of the 29th International Conference on Very Large Databases (VLDB)*, Berlin, September 2003.

[Kan01] Gene Kan. Gnutella. In Andy Oram, editor, *Peer-to-Peer. Harnessing the Power of Disruptive Technologies*, pages 94–122. O'Reilly, Sebastopol, CA, 2001.

[KG03] Peter Z. Kunszt and Leanne P. Guy. *The Open Grid Services Architecture, and Data Grids*, pages 385–407. In Berman et al. [BFH03], 2003.

[Mis04] Miscellaneous authors. Open Grid Services Architecture Data Access and Integration (OGSA-DAI), 2004. http://www.ogsa-dai.org.uk, accessed on 01 June 2004.

[ÖV91] M. Tamer Özsu and Patrick Valduriez. *Principles of Distributed Database Systems*. Prentice-Hall, Englewood Cliffs, New Jersey 07632, USA, 1991.

[SMK$^+$01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the SIGCOMM 2001 conference*, pages 149–160. ACM Press, 2001.

[Wat03] Paul Watson. *Databases and the Grid*, pages 363–384. In Berman et al. [BFH03], 2003.