2012

Semantically-aware data discovery and placement in collaborative computing environments

Xinqi Wang
Louisiana State University and Agricultural and Mechanical College

SEMANTICALLY-AWARE DATA DISCOVERY AND PLACEMENT IN COLLABORATIVE COMPUTING ENVIRONMENTS

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by
Xinqi Wang
B.S., Taiyuan University of Technology, Taiyuan, China, 2002
May, 2012

Acknowledgments

For the following dissertation, I would like to thank several people for their important help along the way. First of all, I would like to thank Dr.Tevfik Kosar and Dr.Jianhua Chen, my Dissertation Chair. Dr.Tevfik Kosar provided the timely and enlightening comments and evaluation at every step of the dissertation process, as well as the direction and guidance to my work upon which this dissertation is based. Without Dr.Tevfik Kosar's help and guidance, it would be impossible for me to complete this dissertation on schedule. Throughout the entire span of my PhD study, Dr.Jianhua Chen's masterful teaching on important courses laid the foundation for my knowledge on Artificial Intelligence, Machine Learning as well as Data Modeling, all of which are important to my works.

Next, I would also like to thank several of my colleagues for their contributing input, chief among them, Mr.Dayong Huang, my colleague and collaborator during the early stage of my work. I would like to thank Mr.Huang for helping me with the initial development of the system. It is also my pleasure to thank CCT (Center for Computation & Technology) for the excellent research environment as well as providing the necessary hardware and science drivers needed for my work.

I would also like to thank Dr. Gabrielle Allen, Dr. Konstantin Busch and Dr.William Adkins for serving on my graduate committee and the help and suggestions, both technical and procedural I received from them along the way.

I would also like to thank Mrs.Sylvia Murphy and Mrs.Cecelia DeLuca of University of Colorado at Boulder and National Center for the Atmospheric Research for their kindly and informative guidance.

In addition to the technical and instrumental assistance above, I would also like to thank my parent, my sisters and all of my friends, both here and abroad, for the support for me, it would not be possible for me to go this far without them on my side.

This project is in part sponsored by the National Science Foundation under award numbers CNS-0846052 (CAREER), CNS-0619843 (PetaShare), OCI-0926701 (Stork) and EPS-0701491 (Cyber-

Tools), and by the Board of Regents, State of Louisiana, under Contract Numbers DOE/LEQSF (2004-07), NSF/LEQSF (2007-10)-CyberRII-01, and LEQSF(2007-12)-ENH-PKSFI-PRS-03.

Table of Contents

List of Tables

List of Figures

Abstract

As the size of scientific datasets and the demand for interdisciplinary collaboration grow in modern science, it becomes imperative that better ways of discovering and placing datasets generated across multiple disciplines be developed to facilitate interdisciplinary scientific research.

For discovering relevant data out of large-scale interdisciplinary datasets. The development and integration of cross-domain metadata is critical as metadata serves as the key guideline for organizing data. To develop and integrate cross-domain metadata management systems in interdisciplinary collaborative computing environment, three key issues need to be addressed: the development of a cross-domain metadata schema; the implementation of a metadata management system based on this schema; the integration of the metadata system into existing distributed computing infrastructure.

Current research in metadata management in distributed computing environment largely focuses on relatively simple schema that lacks the underlying descriptive power to adequately address semantic heterogeneity often found in interdisciplinary science. And current work does not take adequate consideration the issue of scalability in large-scale data management.

Another key issue in data management is data placement, due to the increasing size of scientific datasets, the overhead incurred as a result of transferring data among different nodes also grow into a significant inhibiting factor affecting overall performance. Currently, few data placement strategies take into consideration semantic information concerning data content.

In this dissertation, we propose a cross-domain metadata system in a collaborative distributed computing environment and identify and evaluate key factors and processes involved in a successful cross-domain metadata system with the goal of facilitating data discovery in collaborative environments. This will allow researchers/users to conduct interdisciplinary science in the context of large-scale datasets that will make it easier to access interdisciplinary datasets, reduce barrier to collaboration, reduce cost of future development of similar systems.

We also investigate data placement strategies that involve semantic information about the hardware and network environment as well as domain information in the form of semantic metadata so that semantic locality could be utilized in data placement, that could potentially reduce overhead for accessing large-scale interdisciplinary datasets.

Chapter 1
Introduction

1.1 Introduction

One of the key problems in scientific computing is the interoperability among different data sources produced by different scientific disciplines or tagged by different metadata standards. As collaboration among disciplines and research groups fast become the norm in modern science, management and leveraging of metadata for projects involving cross-domain collaboration has become increasingly urgent. Metadata enables physical data to be effectively discovered, interpreted, evaluated, and processed, introduction of cross-domain metadata management is critical in extending the traditional functionalities traditionally provided by metadata to cover datasets which have become increasingly cross-domain and cross-standards, furthermore, leveraging information provided by metadata could also potentially help alleviate performance issues born out of having to access increasingly large datasets in modern science by intelligently placing datasets to increase locality and reduce overhead inherent in data transfer in distributed environments.

Today, the scientific research community faces new challenges in metadata management as computing environments become increasingly large and complex and science requires more interdisciplinary collaboration. For example, in the Atlas[18] and CMS[23] projects alone, more than 200 institutions from 50 countries use a data collection which increases by around 5 petabytes annually. These large collaborations involve not only domain scientists, but also computer scientists, engineers, and visualization experts who need to access the data to advance research in their own fields. Traditional catalogue based metadata services have limitations in such application scenarios. It is difficult to handle data integration across different domains; management of domain schema evolution often leads to confusion; and performance under peta-scale computing environment is often not satisfactory.

Metadata refers to information about data itself, commonly defined as “data about data”, and it is essential for cross-domain scientific computing. Without proper metadata annotation, the underlying data is meaningless to scientists. And in an interdisciplinary research environment, it is essential for scientists to access data from domains different from his own efficiently and precisely. Traditional domain-specific metadata schema is inadequate in meeting the demand of interdisciplinary collaborative scientific computing. Because of difference in perspective, tradition and terms used, same item might be described completely differently in different domains. Therefore, it is important to establish conceptual and semantic mapping among concepts from different domains to facilitate cross-domain data access. On the other hand, it is unrealistic to expect to establish a completely unified view of everything and reconcile all the differences among all the domains without sacrificing relevance. Metadata schema also need to take into consideration possible future extension to address possible addition of scientific domains. As discussed above, in cross-domain metadata management, proper balance of integration, relevance and extensibility is essential for enabling efficient and precise access to cross-domain data archive.

A successful cross-domain metadata management not only requires a proper metadata schema, it also needs a powerful enough modeling schema to describe the metadata schema in machine-understandable format. Different conceptual modeling schemas are available for building a metadata management service. Controlled vocabulary, schema, and ontology provide an increasing level of description based on agreements concerning the meaning of terms, allowable data hierarchies, and the overall data model.

Based on description logic [39], ontology describes the concepts and relations that can exist for an agent or a community of agents in a given domain. Generally it consists of taxonomic hierarchies of classes and the relations among these classes. Ontology has two key advantages compared with traditional data modeling techniques: first, it has more expressive power than other traditional data model techniques; secondly, efficient reasoners are available to enable discovery of implicit knowledge and perform constraint verification and checking. But its incompatibility with

most of the existing data intensive computing environment presents a problem for implementation. Existing metadata management in distributed computing environment such as iCAT in iRODS [42] offers compatibility with existing infrastructure in scientific computing. But their expressive power is not powerful enough to accommodate what is needed in a cross-domain metadata schema.

Another factor we must take into consideration is the issue of scalability and performance, in peta-scale computing, metadata schema must also be able to scale to peta-scale while still provide reasonably satisfactory performance. A balanced approach is needed in both high-level architecture as well as low level implementation in order to reach the desired scalability and performance targets. For scalability, a distributed, loosely coupled system architecture is often required. For performance, little research has been done with regard to the performance of key parameters of metadata management in peta-scale computing, either for schema rich in expressive power such as ontology, or for schema more traditionally associated with distributed computing. Our experiments [51] indicated that there are unresolved issues in both approaches.

Another untapped application area for metadata is scientific data placement in collaborative distributed computing environments. Current data placement strategy for distributed system, for the most part, simply places data archive physically on the sites physically housing the research project, e.g. Numerical Relativity Group is a Center for Computation&Technology research group at Louisiana State University-Baton Rouge, therefore, data archive for Numerical Relativity Group is created and hosted on cluster located on LSU campus so that spacial locality could be maximally exploited for optimal performance if the researchers of Numerical Relativity Group were to access their data archive. Potential drawbacks for the current data placement strategy include:

1. Due to increasingly distributed nature of modern science, physically concentrating data archive on one location might negatively impact service performance for researchers located in another institutions.

2. The needs for replication to balance availability and performance as well as minimize replication overhead on system are not considered in current strategy.

3. No consideration is granted to potential collaboration among different projects working on similar domains, e.g. gulf coast oil spill simulation might need to work with hurricane prediction group to make accurate assessment.

Semantically-Aware data placement and replication seek to take advantage of user-defined metadata information to optimally place and replicate data across the whole system so that better performance, availability could be provided and overhead could be minimized. Specifically, by utilizing user-defined metadata information, hopefully, data placement and replication strategy can take factors with real-world performance implications, such as semantic locality, to address issues raised above with the ultimate aims of providing service with higher consistence, better performance and higher availability.

In this thesis, we seek to investigate issues related to the development of cross-domain schema, the implementation of cross-domain metadata schema in a collaborative distributed scientific computing environment. We will also conduct tests on key parameters of our systems to better understand the role played by these parameters on performance and scalability. We will also investigate Semantically-Aware data placement strategies and algorithms to leverage cross-domain metadata to improve data locality for better data-accessing performance.

1.2 Summary of Contribution

Large-scale data management has become increasingly centric in modern internet-scale, highly distributed computing environments, as proven by the rise of data-intensive computing in academia and big data in industry, but in particular in industry. Traditional relation-based data model with heavy focus on data consistency epitomized by relational database management system does not provide sufficient horizontal scalability to keep up with the explosive growth of data. Various NoSQL database and key-value stores have been developed to provide the necessary scalability such as Google BigTable[22], Amazon SimpleDB[2] and Apache HBase[3]. These industrial solutions achieve high horizontal scalability and high availability by trading off strict consistency for looser eventual consistency and by embracing key-value stores with little or not schema involved.

They serve well in their respective domains. On the other hand, the problem facing scientific research in academia differs from those facing industry, where most active work on large-scale data management has been conducted, specifically:

1. Industrial solutions are often tailored for specific needs in a controlled environment, or provided as a highly abstract solution that requires substantial redevelopment by users to satisfy their specific requirement. Industrial focus on scalability is concentrated on scaling up the number of data items the system is capable of handling. Scientific management of large datasets, however, often involves highly independent research groups collaborating with each other with little central direction, and due to the nature of present day scientific research, a unique challenge in data management in scientific community is the scaling up of the semantic ”understanding“ of the data being managed, in another word, it is more focused on fusing wildly different data domains so that scientists could understand each other better.

2. The data size in scientific community, while really large, still can not be compared to the data size required in industry, which provides the opportunity for scientific data management to leverage data models whose complexity would lead to overhead that could not be tolerated in industry.

These challenges are unique to scientific community, yet due to the available resources, available datasets, difficulties in understanding and incorporating often implicit relations among different scientific domains, the problems mentioned above have not been adequately addressed inside academia while the explosive growth of data inside academia presents the same set of problems to scientific community as to industry.

In summary, this dissertation seeks to contribute to addressing the overall large-scale data management problem that has become the one of the most important problems in academia and industry alike with specific focus on unique problems and challenges facing scientific community that industrial and open-source solutions of big data problem have not adequately concentrated on. The approach taken by this dissertation is mainly on making different trade-offs among key objectives: scalability, availability performance, etc. to suit the unique needs of scientific community while

still maintaining sufficient attention to providing solution to large-scale data management common to all.

1.3 Dissertation Outline

This dissertation is separated into the following parts:

1. Chapter 2 gives a overview of current status of research in key related areas such as cross-domain semantic modeling, scalability in metadata management and performance evaluation as well as semantically-aware data placement. Chapter 3 provides summaries of key technologies this dissertation leverage.

2. Chapter 4 and Chapter 5 discuss abstract models, metrics and approaches used used in system design and implementation.

3. Chapter 6, 7 and 8 discuss details in design, implementation and evaluation of semantically-aware data discovery.

4. Chapter 9 presents details on design, implementation and evaluation of semantically-aware data placement.

7. Chapter 10 concludes the dissertation, discuss contributions. and future works.

Chapter 2
Background

2.1 Cross-Domain Metadata Management

Metadata management typically deals with defining, representing, storing, accessing properties to be used for content descriptions. Kashyap et al. [34] classified metadata into Content Independent Metadata and Content Dependent Metadata. Content Dependent Metadata can be further categorized into Direct Content-based Metadata, Content-descriptive Metadata. Content-descriptive Metadata comes in two different flavors: Domain Independent Metadata and Domain Dependent Metadata. In interdisciplinary research, the hardest and the most meaningful part of metadata in interdisciplinary collaboration is often Domain Dependent Metadata as it is often hard to describe domain metadata in a clear, structured manner, but it is also the part of metadata information that offers the biggest promise in establishing conceptual mapping among different terms from different domains.

Currently, widespread application of metadata in the management of various entities in various computing environments have produced myriad metadata standards, modeling schemas, etc. describing every kind of metadata. For example, Dublin Core [6] lists 15 key properties such as Title, Creator and Subject that are common elements shared by most entities in most computing environment. However, the success of this kind of minimalistic approach also means it relies too much on textual description, which can be problematic in an interdisciplinary research environment. In an interdisciplinary environment, scientists often need to access data from other scientific domains annotated by metadata described in different vocabulary, which requires metadata to capture the underlying concepts, which in turn, require mapping and integration of terms in different domains based on the similarity of underlying concepts. Projects such as those conducted by Mid-America Heart Institute [41] do support mapping and integration of concepts and terms to capture

the common concepts behind heterogeneous terms, but they operate in single domain environment, while heterogeneity does exist in such environment, it can not be compared to an interdisciplinary environment covering diverse scientific domains.

According to National Center for Supercomputing Applications [25], it is critical to identify communities of scientific data producers and consumers, because most of the current metadata research is conducted within the confinement of single data collection, which is clearly not sufficient in modern interdisciplinary research environment, but the sheer diversity of the entire scientific discipline means that it is not realistic to integrate all the scientific data collections without glossing over significant details, in short, science as a whole and all the players in science, such as scientists, academic institutions, funding agencies, etc. are simply too diverse to be organized and integrated into an single framework at this stage. Also philosophically speaking, it is hard to imagine the existence of a god-like know-it-all entity, no such entity exists in human society, it would be difficult to contemplate the creation and maintenance of such an entity in a conceptualized world whose knowledges and capabilities, so far, all come from its human creators. Fundamentally speaking, the need to "go deep" and the need to "go broad" demand contradictory approaches, it is hard, if not impossible to accommodate both demands to their maximum without introducing unmanageable complexity. Hence, identification of appropriate scientific communities whose data collections are integrable via metadata is a critical first step in the design and implementation of cross-metadata management framework.

Under Grid environment, metadata management is also considered a critical part. iRODS [42] data grid software developed by San Diego Supercomputer Center stores system-related metadata in a central database called iCAT while users can define their own sets of metadata in Subject-Predicate-Object triples format by using iRODS command. This approach only provides anset of tools for users to develop their own metadata without an set of commonly agreed terms, relationships and concepts, as a result, it is very difficult, if not impossible to integrate metadata arbitrarily defined by scientists from different disciplines or even scientists from the same disciplines.

Metadata Catalog Service [44] developed at Information Science Institute, University of Southern California incorporated metadata schemas that describe properties of logical file, logical collection and provenance. Extensive experiments on scalability and performance of MCS under data-intensive environment was also conducted on up to 5 million logical files each associated with 10 attributes. But not enough research in MCS was done regarding building, mapping of terms and concepts of domain-dependent ontology, which can help facilitating interdisciplinary research by unearthing deeper connections among terms and concepts from different domains.

Earth Science Modeling Framework (ESMF) [8] is a set of programming libraries designed to facilitate the building and coupling of earth science simulation modeling. Its metadata system mainly appears in the form of Attribute, here Attribute refers to name-value pairs. Attribute in ESMF can be uniquely identified by its name, its convention and its purpose, sets of Attributes with identical convention and purpose can be grouped together into Attribute packages. Usually Attribute packages describe community standard so that users can denote datasets and other programming objects with terms widely used in their respective science domains. Attribute packages can also be nested inside each other to create basic layered structure. Extensive testing and performance benchmarking under distributed environment have been conducted on ESMF Attribute system, though its simple representation of metadata (name-value pair) and lack of complex structure contribute to better performance in distributed computing environments, they also render cross-domain access of data archive extremely difficult as there is no support for cross-domain semantic mapping of terminologies.

While metadata management is often associated with large-scale scientific datasets, Other areas of computer science touch upon this topic as well. For example, lots of research has been done in the area of Semantic Web [50] whose objective is partially to build an structured, well-defined vocabulary to inject machine-understandable semantic meaning into traditional only human-understandable web pages. Ontology is a modeling scheme used in Semantic Web to explicitly represents a set of concepts within a domain and the relationships among these concepts [30]. It provides rich de-

scriptive capabilities which can be utilized as a metadata modeling scheme to potentially integrate highly heterogeneous data set.

One common use case is the Gene Ontology [29] in which an structured representation of gene functions is used in a uniform way to be queried across different gene databases. Gene Ontology is an important collaborative effort and it is arranged in a hierarchical manner using directed acyclic graph. A controlled vocabulary is provided by analyzing the semantic structures of the data and then implementing a uniform representation of metadata information. The metadata can be queried at different levels over many databases that span the world [29]. Despite its potentials, current research primarily focuses on the development of ontology that defines vocabulary to describe concepts specific to single domain.

Luis E. Bermudez presented an Ontology Metadata Framework in his PhD thesis [36] to facilitate the semantic interoperability among different hydrological metadata specifications. With regard to domain-dependent ontology integration, Stuckenschmidt et al. proposed integrating different domain ontologies for data integration [47] and identified different mapping approaches for concepts in different ontologies.

Jeffrey et al.[32] developed an ontology enabled semantic search engine for the SRB/MCAT [20] system to handle heterogeneous data sources. Their system allows user to load different ontology instance datasets into a mySRB interface enabling user to search on heterogeneous ontology repositories. However, their research did not cover the critical issues of extensibility, limit of extensibility, scalability and performance in a data-intensive environment.

The Pegasus group developed a virtual metadata catalog which provides semantic-rich information for the metadata catalogue [27]. They integrated datasets from three disciplines by constructing one virtual metadata catalog which hides all the underlying distributed domain ontologies from the query mediator. Again, their system didn't adequately address how to integrate metadata of new domains into existing metadata framework with minimal disruption of service. Performance of their system in a terabyte or even petabyte environment was not explored either.

The Earth Science Curator[7] seeks to develop community wide metadata standards. The metadata being developed includes such topics as simulation genealogy, component-level scientific and numerical properties. The Earth Science Curator ontology supports classification of datasets as well as network access of data archives. But the structure of the ontology is essentially flat and the domain covered by it only extends within geo-science community. As a result, there is no support for cross-domain metadata management in ESC and the problems of including it in large-scale computing infrastructure as well as its performance in large-scale, data-intensive computing environment are yet to be resolved.

The CUAHSI Hydrologic Information System (CUAHSI-HIS)[4] provides low-level metadata management in the form of Observation Data Model (ODM), a relational database schema encoded with metadata in Hydrology community; data transmission standard in the form of Water Mark-up Language (WaterML) to hide the underlying heterogeneity related to differences in metadata standard; controlled vocabulary contained in an ontology to guide access to federated, geographically distributed data archives. The CUAHSI-HIS has the most complete metadata management in large-scale distributed environment at this moment, but like the Earth Science Curator and many other systems, it is designed to focus on metadata management in an single domain, attempts to introduce cross-domain management have been initiated but at this moment, it is still at very early stage and progress thus far is limited.

Z. Kaoudi et al [33] designed a p2p-based network for storing, querying and updating RDF metadata describing web or grid resources. It is a web service oriented architecture focusing on describing and accessing service providing nodes. The challenges posed by the existence of large-scale data in the network as well as providing access to not just nodes hosting these data but also access to specific data archive within the nodes are not addressed.

As discussed above, scalability and performance are very important factors in the design and implementation of metadata system in a data-intensive environment. Especially so in modeling schemes rich in expressive power such as ontology. iRODS enabled federated iCAT since ver-

sion 1.2 that can distribute load to multiple iRODS servers to avoid bottleneck, thus increasing performance and scalability. Metadata Catalog Service project did extensive experiment on scalability and performance on query and addition on up to 5 million instances with 10 attributes each deployed on an single node. It did not, however, measure performance and scalability of information-rich metadata which is necessary for domain-dependent metadata integration.

Pan et all. [40] experimented with loading 4,1741 ontology, approximately 45 million triples into relational database in approximately 15 days. While their experiment clearly showed ontology offered the scalability to handle large number of metadata instances needed for peta-scale data grid, the experiment was not expanded to include performance benchmarking on adding, deleting, modifying and querying metadata stores and what, if necessary, needs to be done to overcome performance bottleneck. Also, their experiments did not test scalability and performances on ontology deployed on multiple geographically distributed nodes. Distributed Ontology can increase scalability and flexibility, but it also introduces added complexities of managing the evolution of multiple ontologies and maintaining consistency because local replication is often needed for performance purposes.

A.Maedche and others [38] discussed an integrated framework implemented in KAON [9] for managing distributed ontology in Semantic Web context, their framework adopted the pull approach for synchronizing ontology and replicas residing on different nods and evolution log for updating ontology replicas.

All of the above works have something to contribute to cross-domain distributed metadata management in collaborative computing environment. However, none of them adequately addresses challenges presented by interdisciplinary scientific research often dealing with hundreds of terabytes or even petabytes of data. in this work, we mainly seek to address cross-domain metadata modeling, implementation of cross-domain metadata, benchmarking and evaluation of performance and scalability for real-world usage.

2.2 Semantically-Aware Data Placement

In his dissertation [26], Lei Cao proposed to employ semantically-aware replication to support edge service architecture to achieve optimal mix of Consistency, Availability, Response time and Partition resilience. Semantic-aware replication (SAR) seeks to replicate data based on the awareness of properties of the replicated data so that further access of separate replication node or database service could be minimized. The SAR encapsulate information into different distributed data objects based on application requirement so that different replication strategy could be employed to replicate data to ensure optimal trade-off among Consistency, Availability, Response time and Partition. In the prototype which encapsulates semantic information about an e-commerce system in the form of distributed objects: catalog, order, user profile, inventory and best-seller. Distributed object holds both front end interface, business logic as well as backend data. Different replication strategy is employed to replicate different object, e.g. one to many update is required for catalog object; order object implements the abstraction of the single-reader/multi-writer scenario.

Yu Hua, et al. proposed SmartStore [31], a new generation of distributed semantic-aware file system that enables range query as well as Top-K query with improved performances. Instead of traditional directory-based file system, SmartStore groups file metadata into multiple semantic R-Trees that each represents a ”view” of the distributed data archive based on selected metadata parameters. Metadata represented in R-trees include physical and behavior attributes of files such as access frequency, amount of read and write operations (these two can be grouped together because they both change frequently) filename, creation time (they can be grouped together because they change infrequently), content-based metadata can be grouped according to their level of correlation, such as metadata about files under the same directory could be grouped together because application is more likely to access file under the same directory repeatedly. Because of semantic locality offered by semantic grouping of metadata in R-Tree based data view, SmartStore limits searches to a limited scope of semantically related groups, thus improving system scalability and

reducing performance latency. SmartStore also provides insertion and deletion services so that load on R-Tree nodes could be managed to ensure efficiency and performance.

Hong Tang, et al. adopted Base Scheme [49] in Sorrento, an self-organizing storage cluster, as data placement and migration strategy to balance I/O load and storage usage among distributed nodes. Base Scheme allocates storage based on weight value assigned to available storage providers, weight value of a provide is calculated based on its current workload and available resources. Based on empirical evidence, the storage factor is calculated as the logarithm of the ratio between the available space and the segment size. The load factor is calculated as the inverse of the current workload. The system, upon receiving requests for data placement, will choose storage provider randomly so that each storage provider has a chance of being selected proportional to its weight.

Muthian Sivathanu, et al. developed [46] semantically-smart disk system that leverages higher level information from the file systems to provide better functionalities and improved performance. SDS does not change the interface between file system and disk system, SDC relies on EOF (Extraction of File System) to automatically extract information and layout of file system. Information extracted includes type of blocks (file, directory, etc.), how many and which data blocks constitute a file, etc. By exploiting information extracted, the system also implemented a few functions such as secure delete, structural caching and journaling that are not available under traditional disk system.

Muthian Sivathanu, et al. sought to apply semantically-smart storage technologies to relational database systems.[45] Via log-snooping and explicit access statistics, the system gathers up static and dynamic information about database system to be used by underlying storage system. Information gathered includes block ownership by tables and indices, block type information (tables, indices) as well as information about various access patterns. By implementing an optimistic strategy or a pessimistic strategy, the system deals with observed dynamic information with correctness or performance prioritized respectively. In three case studied, the system implemented three different semantically-smart storage systems, D-GRAID, , FADED, X-RAY, underneath database system to

evaluate if benefits achieved by these semantically-smart storage systems could be achieved under database systems and the cost of such improvements. The results show that semantically-smart storage could be applied to database system with, depending on functionalities, various levels of overhead.

Zhichen Xu, et al. proposed a generic data model to capture the needs of users and applications of semantic-aware file store.[52] The generic data model proposed seeks to be extensible and capable of handling dynamic evolvement of file store semantics. It is based on Resource Description Framework (RDF), namely, subject-predicate-object triple data model. Metadata described in the proposed data model includes: file versioning, hierarchical name space, arbitrary sets of dependencies, associative semantics and context information. The data model also exploits RDF capabilities such as inheritance, namespace, etc to handle changes and evolution of metadata schema during its life time.

Pinar Alper, et al. gave an overview of existing semantic grid middleware in [17]. It discusses semantic grid middleware such as S-SRB, GRIMOIRES, SMDS, etc. Systems examined here were developed mainly to facilitate resource discovery and resource access as well as resource integration via intelligence application of Semantic Web technologies such as ontology and reasoning. There is not sufficient discussion about data placement and corresponding gains in performance. The paper also examines problems related to high-level and abstract view of issues and requirement related to the development of semantically-aware grid middleware.

Sharad Agarwa, et al. developed a data placement system, Volley[43] for cloud services that seeks to decrease inter-datacenter traffic and latency. By exploiting previously unexploited parameters such as network bandwidth among data centers, data inter-dependency, data sharing, etc. Volley incorporate an iterative algorithm that analyzes log files containing IP address, call tree (describing data inter-dependence.), if migration is found to be worthwhile, Volley triggers application specific data migration mechanism. Experiments were conducted on log of Microsoft Live Mesh and Live Messenger services that show significant reduction in inter-data center traffic and latency.

Current strategies for semantically-aware data placement and replication include:

1. By analyzing access behavior, data archives are replicated in such a way that potential users of data have access to replications that are closer to users than the originals. Benefit of this approach is obvious, by exploiting factors that impact data access performance such as network traffic and topology , data users could be directed to use replication that offers the best performance. On the other hand, dynamic analysis of access behavior is needed to extract patterns so that strategies and replications could be updated to reflect the current user environment. E.g. [26]. For example, the UcOMS project archive is hosted on PetaShare under one single virtual collection while UcOMS team is spread throughout LSU Baton Rouge, ULL and Southern University, each with its own PetaShare site, it is conceivable that original archive uploaded at each site should be stored at its local server while a behind-scene service be developed to catalogue access patterns of different sub-collections by different UcOMS groups located at different sites so that a behavior profiles could be in place to facilitate replication, e.g. most frequently remotely accessed files in a week should be replicated to local server, files not frequently accessed by multiple group at different locations should not be replicated to reduce synchronization overhead. Or users could take advantage of the tagging mechanism in PetaShare's metadata management system to tag the file by its possibility of being accessed in the future by other groups so that the system could be informed to perform the necessary replications for performance optimization. Due to the availability of rule-based system in PetaShare, it is conceivable that services that record access behavior by different groups could be developed to provide the necessary profile based on which data placement strategies can be developed and tweaked. For example, the system can detect data collections accessed remotely, if certain threshold were crossed, the data collection in demand can be replicated to servers closest to groups that access it and new data posted into the collection can be placed in serves closest to where it is most in demand.

2. By exploiting semantic locality, in another word, in exploiting the fact that users who are accessing the current data archive are also likely to be interested in accessing data archive of similar

domains, data archives of similar domains can be placed close to each other to optimize performance of repeated access. This approach can potentially improve the performance of repeated accesses, if applied to replications, synchronization overhead could be limited to only part of the system and potentially be reduced. Again, to successfully implement this strategy, dynamic analysis of accessing pattern as well as semantic relations among different data archives are needed. [31] and [43] implement this approach. The key to this approach, like the previous one, is to build up views of the data archive, how to determine if two archives or sub-archives are related, and to what degree they are related, this could be achieved by observing access pattern over a period so that factors such as if files were uploaded to the same collections, if two sub-collections share same keywords, etc. are collected so that a dynamically updatable graph/ontology could be built up containing the semantic relations observed with regard to one particular archive, future data placement and replication could be determined by performing an analysis of the graph/ontology. User groups can also be given direct access to this graph/ontology so that they could directly put metadata terms and relations related to archive into it for future consideration in data placement and replication. Datasets on hurricanes such as hurricane on PetaShare and datasets that might be considered incoming hurricanes impacting, such as the oil spill collection on PetaShare can take advantage of placement strategies based on semantic locality.

3.By exploiting hardware information such as available network bandwidth, I/O bandwidth, available resources, etc. data could be placed and replicated to balance volume of usage and resources available, thus improving performance and increasing throughput. [26] and [49] implement this strategy. Just like the previous two approaches, this approach depends heavily on acquiring the right information and dynamically adjust data placement and replication. Key effort include acquisition of metadata information and more importantly, how to make the decision regarding actions need to be taken to facilitate improved performance and throughput.

4. Overall, to design and implement semantically-aware data placement and replication, metadata about the system such as physical topology, network bandwidth, I/O bandwidth as well as

semantic metadata about data archives need to be collected and analyzed dynamically to produce a overall view of the system, [31] organizes metadata into semantic R-Trees while others such as [52] model context and content metadata with RDF.

To test different strategies, a controlled experimental environment with controlled sets of data collections from controlled set of domains should be set up to test various access behaviors and corresponding placement strategies, such as:

1. If disk and bandwidth resources were sufficient, it is desirable to place data on server local to the institution hosting the research project.

2. If data collection were accessed by multiple groups that belong to the same discipline, depending on access behavior (if one particular group access the data collection frequently or infrequently, the former may warrant a replication job to be performed while the latter may not.) and available resources, certain part of the data collection should be replicated to servers closest to these groups.

3. If two collaborating group opt to share data, relations should be established either by these group or by system itself in the relation group/tree/ontology which will be consulted by the system periodically for dynamically adjusting data placement and replication strategies, upon sensing the existence of relations among groups, changes to the default strategies can be implemented, for example, data posted by these group could be placed on their respective local servers and server local to their respective collaborating partner.

4.If profile of the system, in term of available resources and network bandwidth, for example, a server goes down, or resources on the server becomes scarcer, etc. changes, another set of placement and replication strategies should be put in place to deal with it, for example, certain rarely accessed data can be replicated to other servers to free up resources for new data from local group, or replications of remote resources can be deleted, etc.

5.If certain data collection were accessed by groups of different domains frequently, relations could be automatically added so that data placement strategies from 3 could be performed, e.g. if the cross-group/discipline access were two way, data from both groups should be placed on servers

local to both groups; or data from the group being accessed be placed on servers local to group accessing these data.

Above are some some of the scenarios that could potentially take advantage of semantically-aware data placement and replication to improve performance, as they show, different strategies are available for different scenarios and successful implementation largely depends on the acquisition of semantic metadata information on hardware context, resource availability, access behavior, users profile. Depending on the frequency of change, these metadata can either be developed, hard-coded and changed periodically to reflect changes or be built up dynamically by observing system behavior during a period.

Chapter 3
Overview of Key Technologies

In this section, I will present a brief overview of key technologies employed in my research . Metadata modeling schemes such as ontology, OWL, an ontology representing language [11], Sparql [14], an ontology query language and Protege [12], an ontology development tool are covered. We also give a brief introduction to function and application of iRODS [42] and iCAT. PetaShare, the principle data-intensive distributed infrastructure for the testing and integrating our research work, is also discussed.

3.1 PetaShare

PetaShare [19] is a state-level data sharing cyber-infrastructure effort in Louisiana. It aims to enable collaborative data-intensive research in different application areas such as coastal and environmental modeling, geospatial analysis, bioinformatics, medical imaging, fluid dynamics, petroleum engineering, numerical relativity, and high energy physics. PetaShare manages the low-level distributed data handling issues, such as data migration, replication, data coherence, and metadata management, so that the domain scientists can focus on their own research and the content of the data rather than how to manage it. Currently, there are seven PetaShare sites online across Louisiana: Louisiana State University, University of New Orleans, University of Louisiana at Lafayette, Tulane University, Louisiana State University-Health Science Center at New Orleans, Louisiana Tech University, and Louisiana State University-Shreveport. They are connected to each other via 40Gb/s optical network, called LONI (Louisiana Optical Network Initiative). In total, PetaShare manages 250TB of disk storage and 400TB of tape storage on these sites. PetaShare is a data-aware resource management system. In light of increasing size of scientific datasets, services such as data-aware scheduler, Stork [35], resource allocation, workflow planner and manager are included in PetaShare to facilitate efficient and effective data access. At each PetaShare site,

an iRODS server is deployed, which manages the data on that specific site. Each iRODS server communicates with a central iCAT server that provides a unified name space across all PetaShare sites. The clients can access PetaShare servers via three different interfaces: petashell, petafs, and pcommands. These interfaces allow the injection of data object metadata information (i.e. any keywords describing files in datasets) to iCAT managed data object metadata store whenever a new file is uploaded to any of the PetaShare sites. The physical metadata information (i.e. file size and location information) is inserted to iCAT using the iRODS API. As part of the PetaShare project, works described in this dissertation enable an semantically-enabled metadata management and query system. It provides an extendable metadata framework that gives a unified view over multidisciplinary datasets; The system also provides fast and efficient metadata query services for physically and conceptually distributed data set of peta-scale.

3.2 Ontology and Metadata

Metadata refers to information about data itself, often defined colloquially as “data about data”. For data intensive computing, a well-defined and well-implemented metadata system is essential for scientists to access large datasets distributed across different physical locations and multiple scientific domains. In the context of computer science, the often cited definition for ontology is: “ontology is a specification of conceptualization” [30] . Ontology is a model that explicitly represents a set of concepts within a domain and the relationships among these concepts [30]. One common use case is the Gene Ontology [29] in which a structured representation of gene functions is used in a uniform way to be queried across different databases. A controlled vocabulary is provided by analyzing the semantic structure of the data and then implementing a uniform representation of metadata information. Based on description logic [21] ontology describes the concepts and relations in given domains. It often consists of taxonomic hierarchies of classes, relations among these classes and individuals that belong to one or more of these classes. Comparing to traditional modeling scheme, the main advantages of ontology include:

1. Ontology is theoretically logic-based. As result, automatic inference is supported.

2. Ontology allows relations among concepts to be defined arbitrarily, as a result, expressive power is greatly enhanced. At the same time, ontology still maintains computational decidability. Ontology modeling has the potential to greatly benefit scientific data management.

3.3 OWL, Protege, Jena and Sparql

Since the emergence of ontology as a modeling scheme of great interest to Computer Science researchers, various research teams have developed several ontology representing languages such as SHOE [13], DAML-ONT [5], OWL. Among them, OWL is developed and promoted by World Wide Web Consortium (W3C) as standard language to represent ontology. OWL provides rich sets of axioms that can be used by developers to model concepts and relations of targeted domains. OWL has three sub-languages: OWL Lite, OWL DL and OWL Full. Each sub-language is tailored to meet demands on expressive power and decidability in different application areas. Sparql [14] is a recommended query language developed by W3C for querying ontology-based knowledge stores. Sparql employs SQL-like syntax and supports queries across diverse data sources. Sparql supports extensible testing of parameter constraint and can return result as either list style result set or ontology files. Developed by HP Labs Semantic Web Program, Jena [1]is a Java programming framework for building, modifying, querying ontology. Jena framework includes OWL API, in-memory and persistent storage support and Sparql query engine and is widely used to develop ontology applications. Jena is also adapted by other ontology modeling tools such as Protege as basis of development API. Protege is developed by Stanford Center for Biomedical Informatics Research as a free, open-source graphic development platform for domain modeling with ontology and development of ontology applications. It includes a graphic ontology development tool for domain-modeling and a set of Jena-based APIs for the development of ontology applications. PetaShare's cross-domain metadata management system is primarily developed with Protege, its query system made extensive use of Jena-based Protege API and Sparql query languages.

3.4 iRODS

iRODS (Integrated Rule-Oriented Data System) is a open-source data grid system developed at San Diego Super Computing Center. iROBS is a follow-up system to the widely deployed, also SDSC-developed SRB (Storage Resource Broker) [20] data grid software. iRODS provides a unified namespace to geographically distributed storage system. Under iRODS management, distributed storage system such as those under PetaShare management will present itself to users and applications as a single virtual file system. This virtual file system would manage storage size equaling accumulated size of storage of all 7 PetaShare sites. iRODS includes several modules, those used in PetaShare are listed below:

1. iRODS sever are installed on all seven PetaShare sites, they are responsible for managing local storages (data storage, data movement, data replications, etc), executing commands from clients and handling communication.

2. iCAT (iRODS Metadata Catalog) is the metadata system of iRODS. It is responsible for all metadata related activities in iRODS. Inside iCAT, there are system-defined metadata, mainly related to physical attributes of files and other data entities, users can also make use of iRODS command “imeta” to add metadata arbitrarily defined by users. iCAT is fast and efficient, but lacks the expressive power of ontology-based system. One of the main focus of our research is the integration of elements of ontology-based cross-domain metadata system with data object metadata system we implemented that mainly based on iCAT.

3. iCommand is a set of unix-like commands provided by iRODS to help users access storage managed by iRODS servers. Most of the common functionalities such as listing, creation, deletion and copying of files and other data entities in Unix file system is implemented in iRODS. Among all the commands, “imeta” is provided for access to iCAT. Our data object metadata system is based on a modified version of imeta.

Chapter 4

Semantically-Aware Data Discovery and Placement Model

Fundamentally speaking, the problem this dissertation is seeking to address can be viewed as the introduction of interoperability into collaborative data management system via cross-domain metadata, in particular, this dissertation seeks to leverage cross-domain metadata to address issues of data discovery and placement in collaborative data management.

But how do we transform the above definition of problem into a set of measurable parameters that can be benchmarked and evaluated in practice? To do that, a thorough overview of key system parameters is needed.

4.1 Semantically-Aware Data Discovery

In practice, introducing interoperability into data discovery requires the creation of a cross-domain metadata management system which needs to take into consideration following factors:

$F(M,I,S)$

Here, M, I, S each represents certain aspect of metadata management: M denotes metadata schema; I represents implementation; S factors in performance and scalability. Most important parameters relevant to the overall quality of the system could be categorized into M, I or S. We will explain in detail and discuss the parameters we choose to model the system as well as the relations among the parameters.

In above function, variable M denotes actual metadata used to describe domains, simulation as well as data archives: name of file, name of domain, file extension, factors pertinent to an simulation, etc. It is the most basic element in metadata management, namely, the vocabulary and terminology used for description. M can be further divided into the following components:

Physical attribute metadata M_p, such as name of file, file extension, etc are simple to describe and often unambiguous in its meaning. It also has little direct relationship to the domain the data

entity belongs to. In another word, physical metadata is often independent of domain, it can be captured with a relatively small set of metadata terms organized in flat structure, and every data entity should be uniquely described by this level of metadata.

Direct content-dependent metadata M_c, such as keyword related to the actual content of the data entity is almost boundless if no constraint were imposed. This kind of metadata is content-dependent but not necessarily domain-dependent as similar terms could be utilized to describe object in different domains. As almost all of the data entities need a brief description regarding its content, the description should be kept very concise, one word optimal, to reduce potential extra overhead.

Domain and cross-domain metadata M_d, obviously, this kind of metadata is domain-dependent, on the other hand, it also usually has no direct relationship with the data entities in storage, which means that this level of metadata could be assigned the task of handling, in an structured way, describing terms, relationships in academic domains as well as relationships needed for mapping terms existed in different domains for the purpose of cross-domain access. And it should not be too detail-oriented so avoid dwelling on too much details more relevant to actual data entity.

In previous mentioned formula, variable I denotes actual implementation of metadata schema in a collaborative distributed environment. Factors can potentially impact decisions made regarding implementation include:

Level of distribution I_d, as distributed system is often easier to upgrade and scale.

Implementation compatibility I_c, sometimes, the difficulty and sacrifice in performance and scalability required can be factor in scuttling preferred implementation strategy.

S denotes the overall quality of the system. To determine S, extensive performance and scalability benchmarking need to be conducted. The benchmarking test will involve $(S_i, S_m, S_q, S_o, S_c)$, which represents key performance and scalability benchmarks in a good metadata management system. Represented benchmarks are:

As discussed above, the main goal of this dissertation is the introduction of interoperability into collaborative system via the leveraging of cross-domain metadata system to help data discovery as well as data placement. According to abstract model of metadata system provided above, performance could be measured by benchmarking key performance-related system operations, based on the abstract model, we use the following parameters and metrics to measure system performance:

Metadata insertion performance S_i, we will use triple/second to measure insertion performance.

Metadata modification performance S_m, we will use triple/second to measure modification performance.

Metadata query performance S_q, we will use query/second to measure query performance.

Scalability mainly involves the ability of the system to expand further without seriously compromising performance. According to the abstract system model, there are three level of metadata schema available in the system, in actual implementation, the first two levels of metadata schema will be described by lower level triple-based data model while the domain and cross-domain metadata will be described by higher level ontology-based data model. For the lower level metadata schema, because it is mainly utilized to describe actual data-object available in the system, whose number can easily run up to multi-millions, which can seriously test the system's ability to effectively store, access and query these many triples, as a result, we will use the following parameter to measure scalability on this level.

Data-object metadata scalability S_o, the number of triples available throughout the system, in another word, triple/system will be used to measure data-object metadata scalability.

On the other hand, the higher level domain and cross-domain metadata schema do not describe actual data object in the system, it is mainly used to describe the domains involved as well as establishing relations among related terms in different domains. Scalability here concerns more about the ability of the system to handle domain-wise expansion rather than pure increase of size of metadata schema. Due to the complexities of ontology model, increase in cross-domain metadata,

in another word, increase in ontology can seriously impact the overall performance of the system as ontology query generally takes longer than more basic database query.

Domain and cross-domain metadata scalability S_c, scalability on this level mainly concerns the ability of the system to scale up to include and integrate more science domains without increasing cross-domain ontology to such a size that it seriously degrades system performance.

In sum, metadata management system in collaborative environment can be abstracted into the following description:

$F[(M_p, M_c, M_d), (I_d, I_c), (S_i, , S_m, S_q, S_o, S_c)]$

4.2 Semantically-Aware Data Placement

For data placement strategy, interoperability means leveraging knowledge garnered from metadata management system as well as information about current network conditions and available storage to intelligently place semantically-related datasets closer to their potential users. Assuming semantic-relation has been detected and established in previously discussed data discovery phase, the two factors impacting data placement strategy in a distributed collaborative environment are:

Network throughput S_{ij}, here S_{ij} represents network throughput from node i to node j and node i to node j are two nodes containing semantically-related datasets.

Available disk space on node i D_i, here D_i represents disk space made available by node i for data placement for other non-local datasets.

The goal of the dissertation regarding data placement is to leverage metadata system mentioned above to intelligently place datasets to take advantage the above two factors so that better performance for data access among semantically-related datasets could be yielded.

For data placement, the problem this thesis seeks to address is to, for datasets semantically-related to users of node i, achieve $Min(\frac{D_{j1}}{S_{ij1}} + \frac{D_{j2}}{S_{ij2}} ..... + \frac{D_{jm}}{S_{ijm}})$, here node j1 to jm represent candidate nodes chosen for data placement, ordered by preference. If available space on node $j1$ had enough disk space for placing dataset semantically-related to users on node i. $j1$ would be chosen as the

sole data placement node. If not, node j2, j3, j3, to jm would be chosen accordingly until all related dataset were placed.

Chapter 5
Methodology

For data discovery, the key objective is to achieve cross-domain data access in metadata management, to achieve "cross-domain" in metadata management in any meaningful sense, extensive amount of communication needs to be conducted with domain scientists to assess and evaluate the practicality of establishing conceptual and/or semantic links between different domains. For domains that can be linked together, metadata schema should be developed in various encoding formats to accommodate different requirement in different context. For example, overly complicated metadata modeling is not practical in the context of large, distributed datasets, while scientists need some kind of controlled vocabulary to guide them through the query process, especially if they were trying to access data of another domain. As a result, we need to develop a higher level metadata vocabulary that is not dependent on any specific implementation. The current metadata vocabulary we have include four specific science domains and we have implemented it in ontology format, with it in place, the ontology can serve as controlled vocabulary to guide users through the query process. We also plan to implement the vocabulary in another, less complicated, closer to infrastructure, format to enable more detailed and direct access. Finally, tests need to be done to thoroughly evaluate the performance and scalability of our system.

For data placement, the key objective is to place datasets that are semantically-related, as determined by cross-domain metadata management system, closer to each other physically so that locality could be maximumly exploited to reduce overhead involved in transferring datasets among geographically distributed storage nodes. We will identify key factors needed for semantically-aware data placement, experimenting with different algorithms and conducting performance benchmarking.

5.1 Metadata Schema

We first design a high level metadata vocabulary that covers multiple domains. The domains covered in our current design include:

Coastal Science and Hurricane Predication, specifically, datasets generated by the SCOOP project.

Numerical Relativity and Astrophysics, specifically, datasets generated by the NumRel project.

Petroleum Engineering, specifically, datasets generated by the UCoMS project.

Scientific Visualization, specifically, visualized files generated from DMA project.

The metadata schema can be roughly separated into three distinct levels:

Data object metadata schema, metadata that describes attributes of the data object and content of data object.

Domain controlled vocabulary, logical relations among terms in domain controlled vocabulary such as equal, subsume, subsumed-by.

Cross-domain concepts mapping, high-level terminology mapping out the overall specification of conceptualization in domains involved.

5.2 Multiple Implementation for Multiple Needs and Environments

Our experiments on current implemented systems indicate that encoding over-complicated metadata to the level of individual file will entail severe degradation of performance and scalability. As a result, multiple implementations of the metadata vocabulary geared toward addressing different concerns of different environments should be developed to ensure acceptable performance. Different level of details and cross-domain connectivity corresponding to the needs and requirement of different implementations should be discussed to strike the right balance between expressiveness and usableness. By implementing metadata management system in a layered approach similar in concept to traditional memory hierarchy, as illustrated in Figure 5.1, we seek to establish a metadata hierarchy in the system so that different level of metadata query could be handled by different

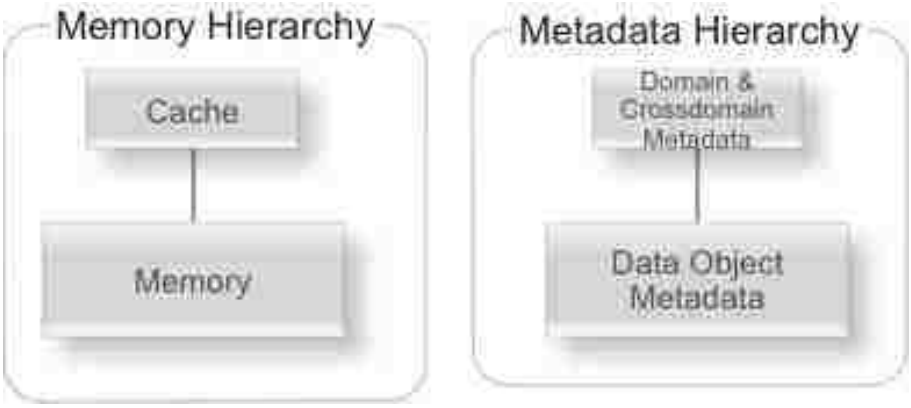

FIGURE 5.1: Metadata Hierarchy

layers in the metadata hierarchy. We currently envision two layers of metadata in the implementation of metadata hierarchy:

Domain and cross-domain metadata access implemented in ontology format, this implementation should mainly cover with domain-level metadata that is used to describe the domain itself. Expression of logical relations that establishes conceptual relationships among terms used to describe domains should also be implemented at this level. Components cross-domain metadata store and domain metadata store in Figure 5.2 illustrate the role of domain and cross-domain metadata played in the overall metadata management architecture.

Data Object metadata access implementation that encodes individual files and folders with necessary metadata informations for more precise access to data archive, this implementation emphasizes lower level description of metadata information needed for individual or small batched access of data archives. Component federated data-object metadata store in Figure 5.2 illustrates the role of data object-level metadata played in the overall metadata management architecture.

5.3 Performance Evaluation

For the last stage, we will conduct thorough and systematic testing on parameters identified in previous section to evaluate the performance and scalability.

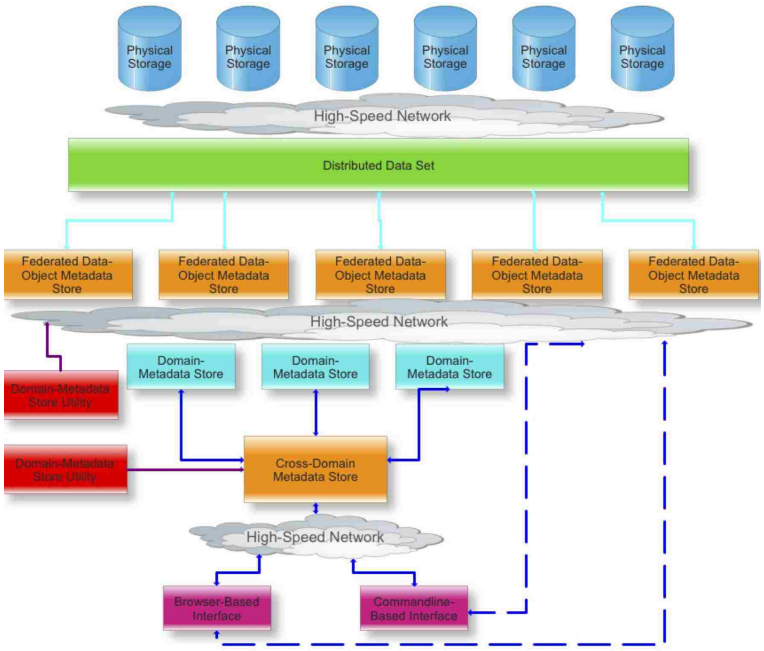

FIGURE 5.2: Proposed Implementation Architecture

5.4 Semantically-Aware Data Placement

Traditionally, data placement algorithms, whether as part of file systems in operation systems, or as part of distributed system, only leverage properties of data file itself and hardware storages, e.g. size of files and size of pages on hardware. Without taking advantage of semantic information concerning the content and context of data, traditional data placement algorithms could potentially introduce unnecessary overhead in a distributed collaborative computing environment as in modern collaborative research environment, the importance of semantic locality often requires placing datasets with awareness of semantical closeness. Another factor to be considered is network conditions such as network traffic and network throughput as intelligent data placement of semantically related datasets need to be aware of the primary factors impacting efficient access of data on remote storage, in another word, close semantic relationship requires dataset to be intelligently placed so that physical closeness among datasets matches their respective semantic closeness. Good semantically-aware data placement algorithms need to take into consideration the following factors:

1. Semantical relationship among terms used to tag datasets. As discussed in previous section regarding data discovery using semantically-ware metadata, this relationship can be explicitly marked by researchers; and be generated based on logic reasoning automatically by machines. For the sake of simplification, we chose metadata standards from WQX (Water Quality eXchange) and National Water Information System (NWIS) used by Environmental Protection Agency (EPA), US Geological Survey (USGS) respectively to tag hydrologic data collected by these agencies.

2.Physical closeness information among different nodes of the distributed system, in our experiments, we primarily measure network throughput in data transfer rate per second as well as available storage on different nodes. The idea is to measure these parameters periodically, then create an set of candidate nodes upon which semantically-related datasets are to be placed so that physical distance among dataset could be closely matched to their respective semantic relationship, thus reducing the overhead for semantically-related groups to access each other's datasets.

.

Chapter 6
Semantically-Aware Data Discovery[1]

6.1 Cross-Domain Metadata Schema

We have developed a ontology consists of metadata about data archives from four science projects in Center for Computation& Technology (CCT), they are SCOOP, Numrel, DMA and UCoMS archives. The structure of this ontology follows a general-to-specific conceptualization with general concepts such as File, Archive serves as common concepts connecting metadata belonging to different science projects.

Currently, our ontology-based cross-domain metadata schema incorporates domain-independent, domain-dependent and provenance metadata of four science drivers: coastal hazard protection (SCOOP) [15], reservoir uncertainty analysis (UCoMS) [10], numerical relativity (NumRel) [16] as well as scientific visualization (Digital Media Archive - DMA) at Center for Computation and Technology, LSU. The following are brief introductions of the four current guiding application scenarios:

Coastal Modeling - SCOOP Archive. The SURA Coastal Ocean Observing and Prediction (SCOOP) program is building a modeling and observation cyber-infrastructure to provide new enabling tools for a virtual community of coastal researchers. Two goals of the project are to enable effective and rapid fusion of observed oceanographic data with numerical models and to facilitate the rapid dissemination of information to operational, scientific, and public or private users [37]. As part of the SCOOP program, the team at LSU has built an archive to store simulation and observational data sets. Currently the archive contains around 300,000 data files with a total size of around 7 Terabytes. Three main types of data files are held in the archive: wind files; surge (water height) files; and data model files. The basic metadata information for these files are: the file type,

[1]Reprinted by permission of "International Journal of Grid and Utility Computing", published by Inderscience Publishers

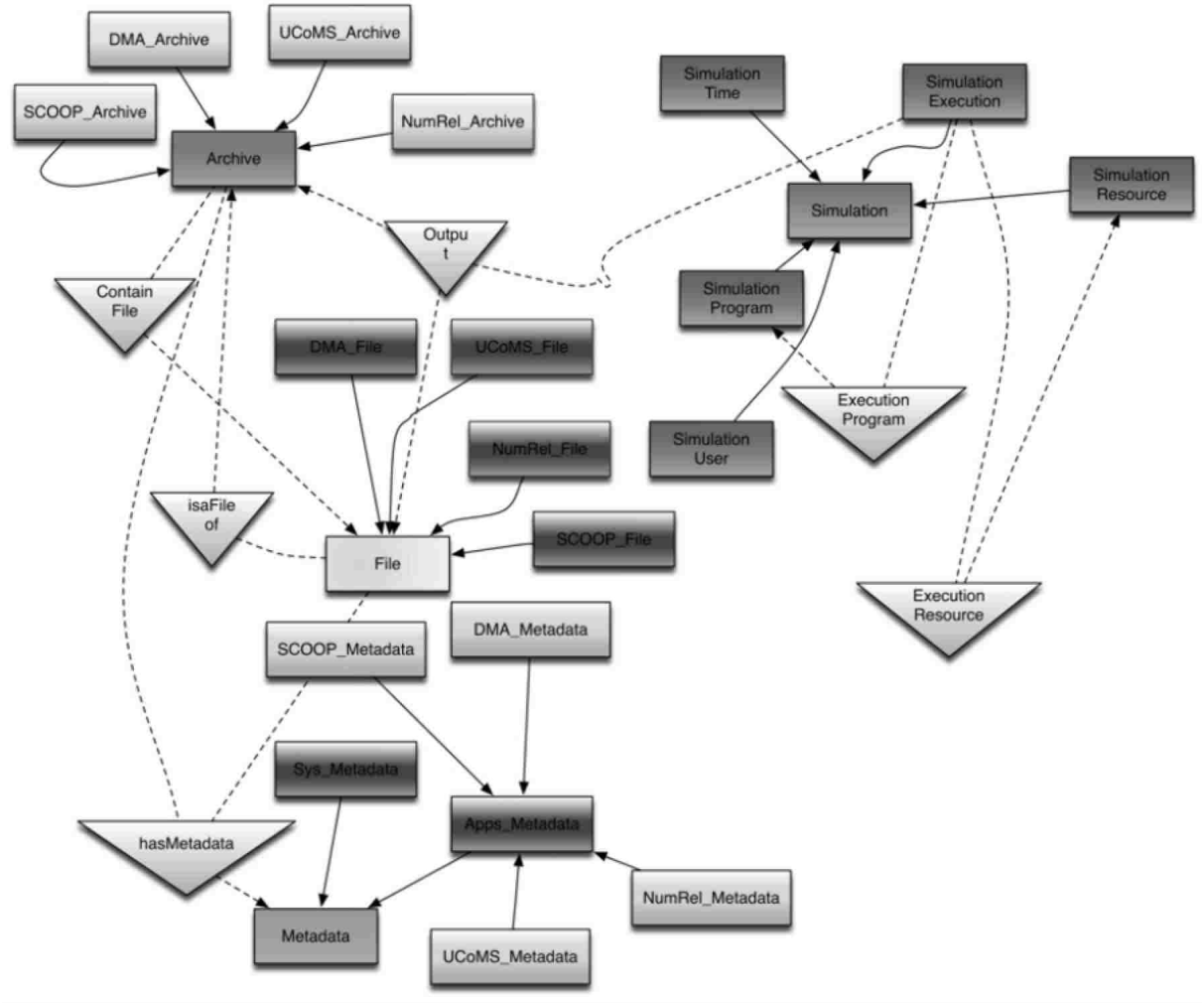

FIGURE 6.1: Cross-Domain Metadata Schema

the model used to generate the file, the institution where the file was generated, the starting and ending date for the data, and other model related information.

Astrophysics - NumRel Archive The Numerical Relativity group at LSU is building an archive of simulation data generated by black hole models. One of the motivations is to analyze experimental data from gravitational wave detectors such as LIGO. These simulations are typical of many other science and engineering applications using finite element or finite difference methods to solve systems of partial difference equations. The simulations often take many CPU hours on large supercomputers and generate huge volumes of data. Software packages such as Cactus [28] enables scientists to develop their code in a modular fashion. Each numerical library in the package

defines a set of attribute names which can be used as controlled vocabulary. The attribute names could describe input parameters or computation flags. Such information is crucial for user's later retrieval.

Petroleum Engineering - UCoMS Archive. Reservoir simulations in petroleum engineering are used to predict oil reservoir performance. This often requires parameter sweeping, where large numbers (thousands) or runs are performed.

In this scenario, users need to provide the initial range of parameter settings. In such a setting, the important metadata can be expressed as follows: parameter name; the range of the parameter in the simulation; the particular parameter value which is set for the run.

Visualization - DMA Archive. Scientific data, after being generated by simulations, needs to be further analyzed. One important tool to help scientists is visualization. The Digital Media Archive (DMA) at CCT is being built to store the resulting images from scientific visualization, along with other media such as movies, sound tracks, and associated information. Visualization metadata can be fairly simple: Image Name, Image Size, Image Width, Image Height, and File Format.

As evidenced by the projects involved, our metadata schema currently covers multiple scientific domains. Metadata described in our metadata schema spans simple domain-independent metadata such as file type (txt, jpg, png, etc.), location (physical location or logical location) and file size, domain-dependent metadata such as different observation in SCOOP (SURGE, WIND or Trans), drilling and reservoir metadata in UCoMS, as well as provenance metadata that describes the steps involved in the generation of data file. We also map files from different domains via content-describing metadata such as the mapping of hurricane observation data from SCOOP to data visualization produced by DMA. Our metadata schema is modeled as a taxonomy of classes, instances and properties connected through relations such as subClassOf, equivalent-to and disjoint-with. Figure 6.1 describes the classes, properties and relations available in our current schema for the description of aforementioned science domains.

Another advantage our metadata schema holds over traditional metadata schema is its potential logical reasoning capability thanks to strong logic foundation upon which ontology is based. Being able to perform logical reasoning on ontology not only provides domain experts and system developers with a necessary tool to check and verify logical consistency, but also increases scalability. In our system, we choose SWRL [48], submitted by the National Research Council of Canada, Network Inference and Stanford University, to represent logical constraint metadata schema has to follow. SWRL provides a high-level abstract syntax for Horn-like rules in OWL DL and OWL Lite.

$$Rule1: SCOOP\_File(?x) \wedge Name(?x,?y) \wedge swrlb:startsWith(?y,"S") \longrightarrow SurgeExisted(?x,true)$$
$$Rule2: SCOOP\_File(?x) \wedge Name(?x,?y) \wedge swrlb:startsWith(?y,"W") \longrightarrow WindExisted(?x,false)$$
$$Rule3: SurgeExisted(?x,true) \wedge WindExisted(?y,false) \wedge Name(?x,?z) \wedge Name(?y,?r) \wedge swrlb:contains(?z,?r) \longrightarrow isValid(?x,true)$$

FIGURE 6.2: Rules for logical inference in Ontology

In our current implementation, there are several scenarios under which logical inference will be needed to ensure the consistency of the ontology. For instance, in the SCOOP ontology, each instance of a surge file requires a corresponding wind file to ensure its validity. According to the naming convention agreed upon by participants of the SCOOP project, the name of the surge file begins with ”S”; and name of the wind file begins with ”W”. At the same time, the file name of the surge file contains the file name of its corresponding wind file. For example, if there is a surge file named SWW3LLFNBIO_WANAFe01-UFL_20050825T0000_20050826T1300_20050826T1300_12hsT272_Z.txt, the name of its corresponding wind file would be WANAFe01-UFL_20050825T0000_20050826T1300_20050826T T272_Z.txt. The three SWRL rules shown in were written in Figure.6.2, upon processing by ontology reasoner, will be used to check the validity of surge files, thus ensuring the logical consistency of the SCOOP ontology.

6.2 Semantically-Aware Metadata System

Currently, we have implemented two metadata management and query systems in based on Protege [12] and iRODS respectively.

As the current de facto standard of ontology design, Protege provides a complete set of tools in support of the design and implementation of ontology-based systems. But the complex nature of ontology and the implementation of Protege turn out to be inadequate to provide sufficient performance.

iRODS, on the other hand, provides its own native metadata system, but the triple-based metadata representation is not powerful enough to satisfy the need for cross-domain metadata management.

Both systems provide some unique advantages the other system currently does not support while at the same time, both systems turn out to be inadequate on other fronts. Based on the model introduced in Chapter 5, two layers of metadata management systems are developed to handle cross-domain and data object metadata management respectively. Protege is chosen as the platform upon which ontology-based cross-domain metadata schema is leveraged to provide cross-domain data discovery capabilities while iRODS is chosen as the platform where data object metadata is used to provide fine-grain query capabilties to the system.

6.2.1 Cross-Domain Metadata Management

The reason we chose to implement the upper layer of our metadata system based on Protege-API and Protege-based database back-end is to take advantage of the semantic expressive power of ontology. As the de facto standard for ontology design, Protege supports almost all the W3C standards and provides support for the whole range of ontology related functionalities, from graphic ontology design interface to built-in reasoner all the way to ontology serialization into relational database, which make Protege and Protege-related technologies good candidates for implementing an semantic enabled cross-domain metadata system.

As shown in Figure 6.3, two different interfaces are available in our system. They are browser-based and command-line-based respectively. The purpose of browser-based metadata interface is to provide an easy-to-use, easy-to-understand method of access so that scientists can query and obtain small numbers of experimental files across multiple domains, while command-line-based

interface can be combined with scripts and other programming tools so that more flexible, more powerful access to bulk files is also available in our system.

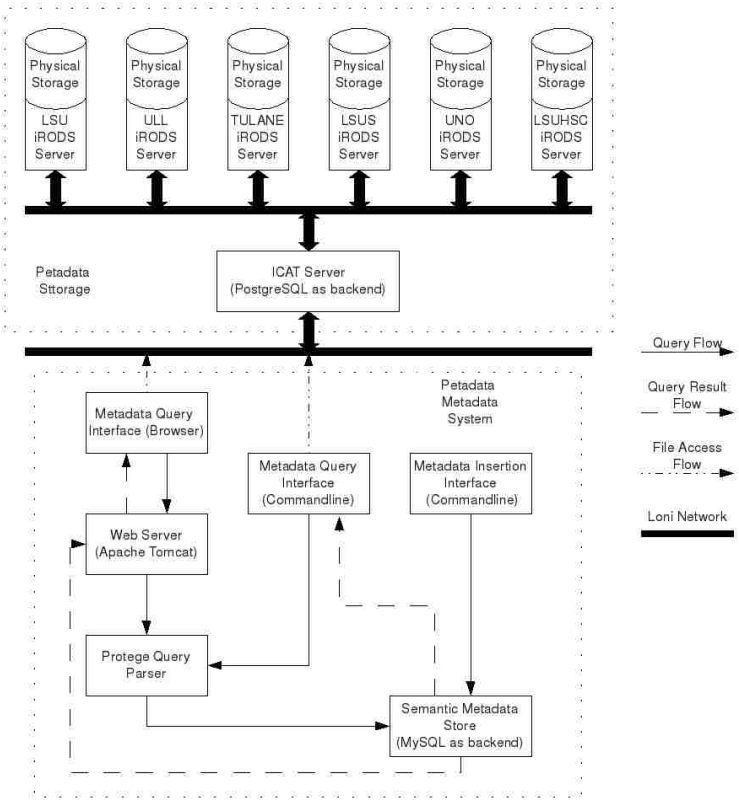

FIGURE 6.3: Cross-Domain Metadata Management System

The core of our system are Protege Query Parser and Semantic Metadata Store. Protege Query Parser is implemented to parse queries entered by users into Sparql [14] queries understandable to Protege query engine. In Semantic Metadata Store, metadata definitions in the forms of ontological classes and ontological instances are stored. Protege itself provides two ways of storing ontology: file-based and relational database-based. The first approach essentially stores ontological classes and instance definitions to text file, although it is easier to implement and access file-based ontology, our experiment showed that file-based ontology can not scale to satisfy the data intensive requirements of modern collaborative science, attempts to insert metadata instances in excess of ten thousands resulted in insufficient memory error. Even though increase of physical memory size can partially alleviate this problem, the fact that Java Virtual Machine places limit on the amount

of physical memory it can handle means text-based ontology can not scale as much as we want. Another problem is it often takes more than a dozen hours to load text-based ontology with more than ten thousands instances into memory. The causes of the failure to scale include:

1. The amount of memory required exceeds the maximum memory size Java Virtual Machine is capable of handling.

2. System is saddled with too high a performance overhead as a result of large numbers of file accesses.

To overcome the above mentioned problems, we decided to take advantage of the second approach and store our ontology in regular relational databases, in our system, we chose MySQL as the back end database in which all cross-domain metadata are stored in ontological form.

Another part of our system is called Metadata-insertion interface. It is a Java-based command-line program that can be utilized, with the help of script languages such as perl, to automatically insert metadata about newly created experiment files.

For example, in large science science experiments, when an experiment file is created, metadata-insertion interface can be triggered to automatically add appropriate metadata information, such as name, keyword, time of creation, file type, etc, into Semantic Metadata Store. The system administrator can also choose to do bulk-inserting, as of now, we have successfully inserted metadata about more than 1 million files.

Cross-Domain Metadata System offers support for ontology-based metadata query, ontology-based automatic metadata insertion, as well as ontology-based file access through both browser and command-line interfaces.

One typical use scenario is:

Assuming a meteorologist needs some monitoring data on Hurricane Katrina's path of movement, he also would like to see visualized pictures of the monitoring data. In real life, raw monitoring and visualized data could belong to different project, different projects may have differ-

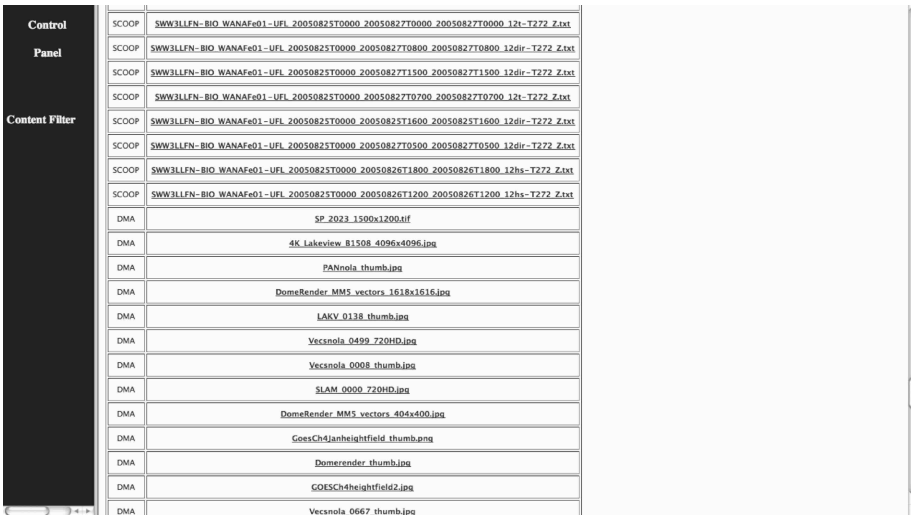

FIGURE 6.4: Cross-domain query result

ent vocabulary for describing data. The use of ontology in Cross-Domain Metadata Management system can bridge the semantic differences that may exist among different science projects. We assume here that raw and visualized data belong to different projects. In this use scenario, on our

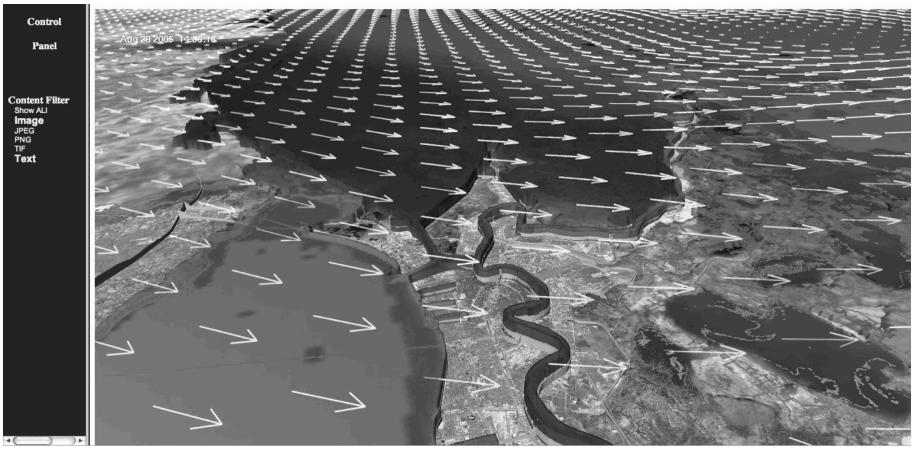

FIGURE 6.5: Data fetched from Distributed Storage

system, the meteorologist could simply open his web browser or the specific command-line interface. Here we assume he chooses the web browser route. Type ”Katrina“ into the search box, and press the search button. Straight arrow in Figure 6.3 shows the data flow of his query. Cross-Domain Metadata Management System will then search its metadata store and return a list of files from both projects it thinks are related to Hurricane Katrina, as indicated in Figure 6.3 by dotted

arrows. Then the meteorologist can simply click whatever file he wants to obtain, the metadata system will send out request to actual storage of these files to fetch the file back into the machine of the meteorologist. Figure 6.4 and Figure 6.5 illustrate query result and file fetched back from remote storage respectively in our currently implementation. Typical workflow involved in query operation is illustrated in Figure 6.6.

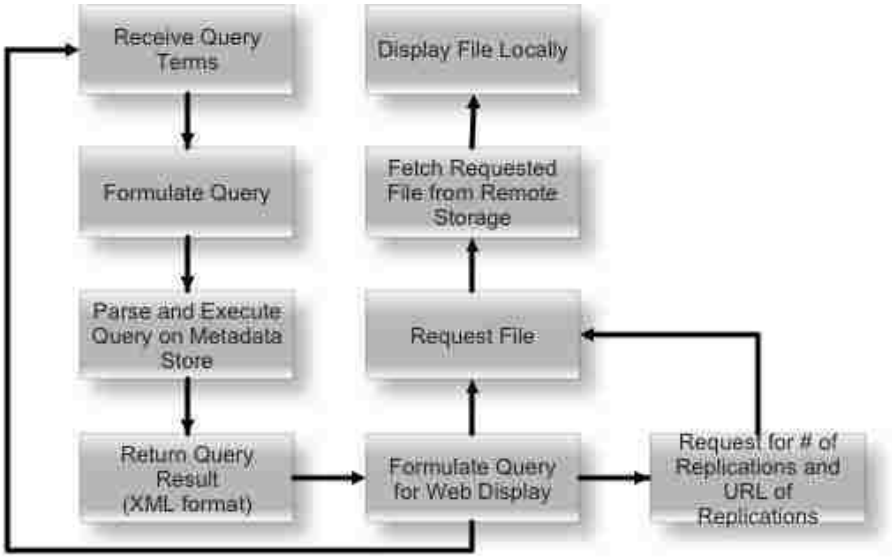

FIGURE 6.6: Workflow in Cross-Domain Metadata Management System

The biggest advantage for Cross-Domain Metadata Management System is the establishment of a unified view of scientific data across different science projects or even different science disciplines. A unified data view can enable scientists to access data from multiple projects from multiple disciplines, regardless of the differences in vocabulary. Such data view is critical in modern, increasingly cross-disciplinary collaborative science.

Shortcoming of Cross-Domain Metadata Management System is clearly illustrated in Figure 6.3. Basically, in exchange for the expressive power of ontology, we have to build another metadata system independent of the iCAT [42] metadata system used by iRODS to provide fine-grain data object metadata management. Doubtlessly, the extra set of metadata and everything related to its management add overhead to overall performance of overall system. Also almost the entire set of

technologies we employed to implement the system is Java-based, which introduces more overhead to performance and more complications to achieve maximum scalability.

6.2.2 Data Object Metadata Management System

Unlike Cross-Domain Metadata Management System based on ontology, iRODS-based Data Object Metadata System does not support a richly representative scheme, namely ontology, like Cross-Domain Metadata Management System does. On the other hand, iRODS and its corresponding iCAT metadata system serve as the backbone of our system. As a result, metadata system based on iRODS and iCAT is naturally integrated seamlessly. Also, unlike ontology technology which is Java-based and was originally designed for Semantic Web with little prior consideration for performance, iRODS and its corresponding metadata system iCAT were designed with the requirements of data-intensive computing in mind. Better performance can be achieved as a result.

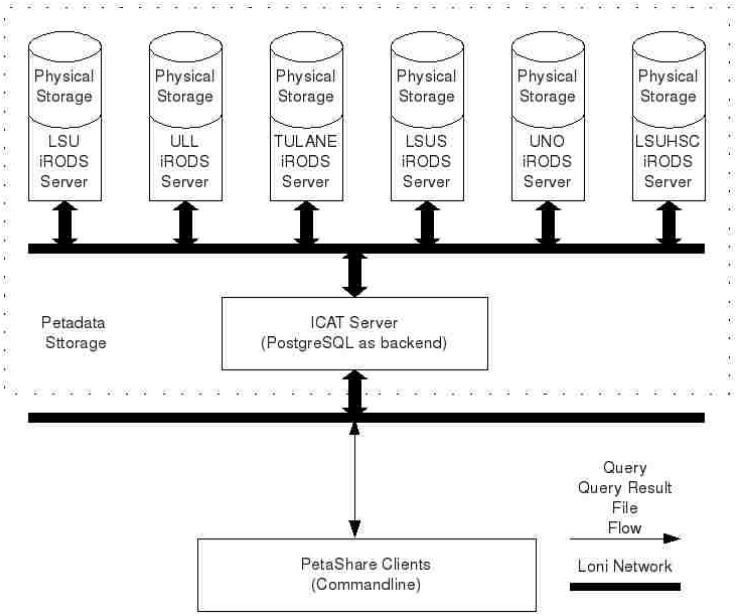

FIGURE 6.7: iRODS-based Data Object Metadata System

As Figure 6.7 shows, the framework of iRODS-based Data Object Metadata System is far simpler. Only one extra layer of system is added to the existing iRODS-based distributed storage. Clients have been developed to parse and remote-execute various iRODS commands. One such

command is “imeta”, which is used for inserting and accessing metadata stored in iCAT. Detailed documentation for imeta can be seen at [42].

Command “imeta” can be used to insert metadata about iRODS files, collections, resources and users in the form of Attribute-Value-Unit triples (AVUs) Because iCAT also employs relational database as back end storage and the fact that iCAT deals with metadata far less expressive than ontology does, we expect it to be able to be at least as scalable as ontology-based Cross-Domain Metadata Management System. Our experiment indicates that iCAT can easily handle file metadata in the order of millions of files. In current implementation, command “imeta” can only insert metadata one AVU at a time. To expand its functionalities, we implemented another version of command “imeta” that supports bulk-insertion function similar to the one provided by Metadata-insertion interface in Cross-Domain Metadata Management System.

In iRODS-based Data Object Metadata System, a typical query operation would be users typing in what they want to query as parameters of command “imeta”. “imeta” will do the query and return a list of files, users then can use other iRODS commands supported by clients to access the files needed to be accessed. Figure 6.8 illustrates the workflow.

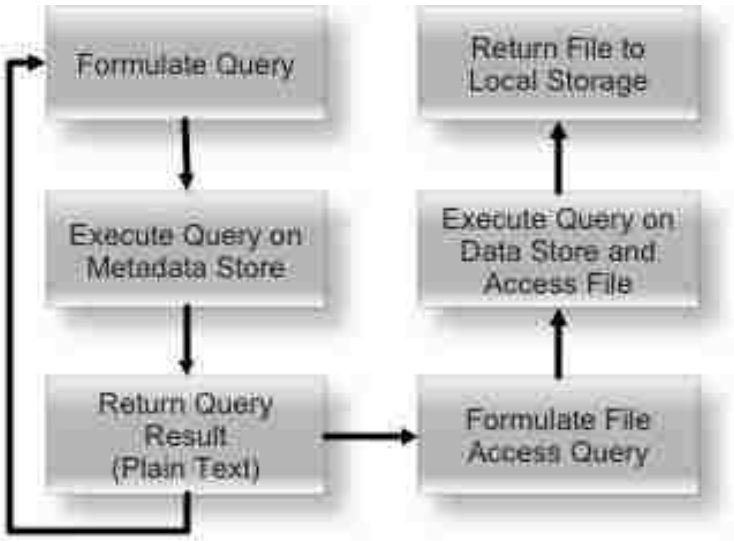

FIGURE 6.8: Workflow in iRODS-based Data Object Metadata System

Chapter 7

Cross-Domain Data Discovery Performance and Scalability Evaluation[1]

We have done performance and scalability benchmarking of Cross-Domain Metadata Management System we built. The performance and scalability experiments are based on SCOOP data archive. The purpose of the SCOOP project is to promote the effective and rapid fusion of observed oceanographic data with numerical models and to facilitate the rapid dissemination of information to operational, scientific, and public or private users [37]. To support SCOOP applications, the team at LSU built a SCOOP archive which stores the related data sets. Currently it contains around 300,000 data files with a total size of around 7 Terabytes.

As discussed and illustrated in previous chapters, the two layer of metadata systems we built offer different level of capabilities and performances. In this chapter, we will seek to investigate the performances and scalabilities of Cross-Domain Metadata System based on the parameters laid out in the model presented in previous chapter, namely, to determine performance S, extensive performance and scalability benchmarking need to be conducted. The benchmarking test will involve $(S_i, S_m, S_q, S_o, S_c)$, these parameters represent:

Metadata insertion performance S_i

Metadata modification performance S_m

Metadata query performance S_q

Data-object metadata scalability S_o

Domain and cross-domain metadata scalability S_c

In this chapter, tests will focus on performance comparisons of (S_i, S_d, S_m, S_q) between native iRODS system and Cross-Domain Metadata Management System as well as S_c, which represents scalability of Cross-Domain Data Discovery.

[1]Reprinted by permission of "International Journal of Grid and Utility Computing", published by Inderscience Publishers

7.1 Performance Evaluation on Cross-Domain Metadata Insertion

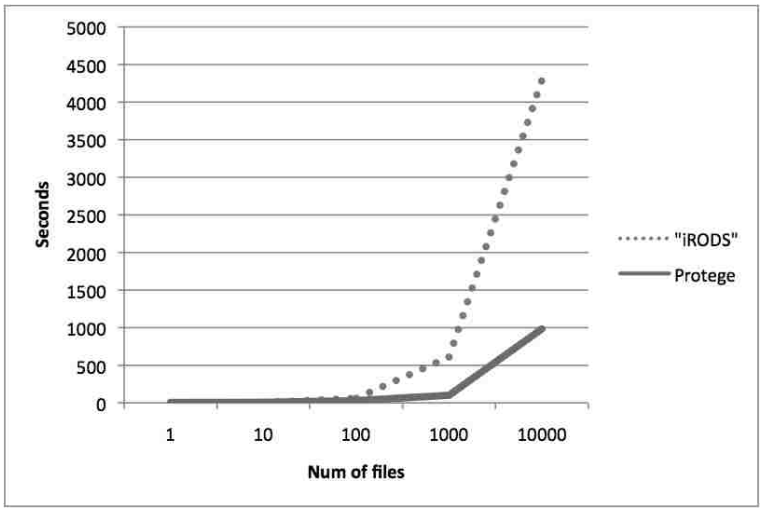

FIGURE 7.1: Performance Comparison of Metadata Insertion

We picked a test case involving the insertion of from 1 to 10000 sets of metadata corresponding to 1 to 10000 experiment files produced by the SCOOP project. The program used in this experiment are bulk insertion program we developed for Cross-Domain Metadata Management system and modified “imeta” command for iRODS. The experiments were conducted on a Dell Desktop with a 2.40 GHz CPU, 512 M memory and Ubuntu linux installed.

As shown in Figure 7.1, as the size of insertion metadata set grows, Cross-Domain Metadata Management System displays far superior performance than the system iRODS [42] has, the performance discrepancy turned out to be a surprise for us as ontology-based Cross-Domain Metadata Management system is required to handle semantically far more complicated data. Our preliminary conclusion is that the iRODS [42] system handles metadata insertion by repeatedly inserting triples into databases, while Cross-Domain Metadata Management System based on existing ontology tools, namely Protege database, handles large set of metadata insertion by bundling them together in the memory, then bulk-inserts them into the database.

7.2 Performance Evaluation on Cross-Domain Metadata Query

On the same testbed we used for evaluating performance of Cross-Domain Metadata Insertion. We formulated queries for both systems that would return result sets with size ranging from 1 to 10000 files. The query we used in our experiment seeks to return files in SCOOP project archive that are related to Hurricane Katrina, here in this example, we assumed that all files created between 00:00:00 08/23/2005 to 23:59:59 08/29/2005 to be Katrina-related files. This query can be finished by executing one Sparql [14] query on Cross-Domain Metadata System or one iRODS command on iRODS system.

Our experiment result indicates, as illustrated by Figure 7.2, that very significant performance gap exists between Cross-Domain Metadata Management System and iRODS system. As shown by Figure 7.2, query performance of iRODS system is in the order of seconds, while query performance of Protege-based system is in the order of hundreds of seconds. On the other hand, query

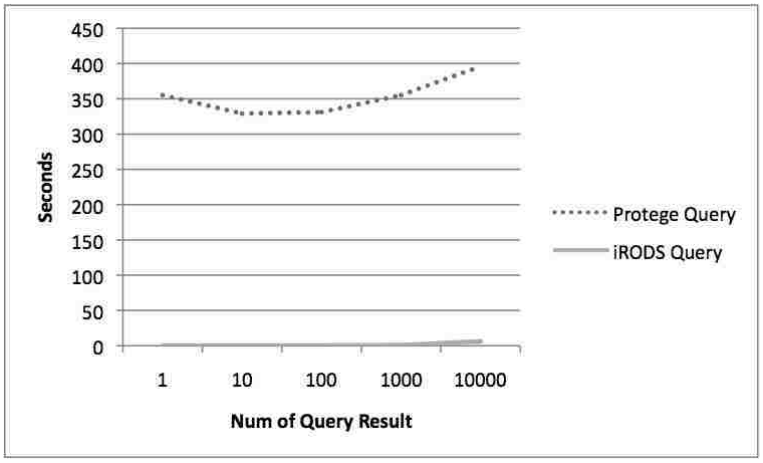

FIGURE 7.2: Performance Comparison of Metadata Query

time on iRODS system is positively correlated to the size of result set, as the size of the result set grew, significantly more time was needed for the query to finish. In the case of Cross-Domain Metadata Management System, however, relatively little performance fluctuations appeared among query result sets of significant size differences. Coupled with our observation that in Cross-Domain

Metadata Management System, most time was spent on execution of query while in iRODS system, most time was spent on parsing and printing of query result, it appears that the performance gap between the two systems can be largely attributed to the far more complicated and rich metadata representation in Cross-Domain Metadata Management System, even though in both systems, metadata is ultimately stored in open source relational databases. (In Cross-Domain Metadata Management System, we adopted MySQL as database back end; In iRODS system, PostgreSQL is used to store metadata; The two database system were chosen because they were the best supported relational database system by Protege [12] and iRODS [42] respectively.)

Another observations of ours was that in Cross-Domain Metadata Management System, query that would return tens of thousands of files often collapsed the system, and when it succeeded, the performance was extremely bad, which indicated that the size of available memory that can be utilized by Java Virtual Machine is also a contributing factor to the far worse query performance by Cross-Domain Metadata Management System.

7.3 Performance Evaluation on Cross-Domain Metadata Modification

In this section, we attempted to test Cross-Domain Metadata Modification performance. Experiment environment remains the same as described in two previous sections. The metadata management task we sought to benchmark this time is the modification of value of metadata ”Keyword” in both systems, the systems would try to modify value of ”Keyword” from void to ”Katrina” based on the time the file was created. Size of files whose metadata are to be modified by the two systems ranges from 1 to 10000.

As illustrated in Figure 7.3, when the amount of files systems tried to modify was relatively small, from a few files to a few hundreds of files, performance of iRODS system is vastly superior to that of the Cross-Domain Metadata Management System. Performance gap in this case is comparable to the performance gap iRODS system achieved against Cross-Domain Metadata Management System in previous section, which is consistent with performance advantages iRODS

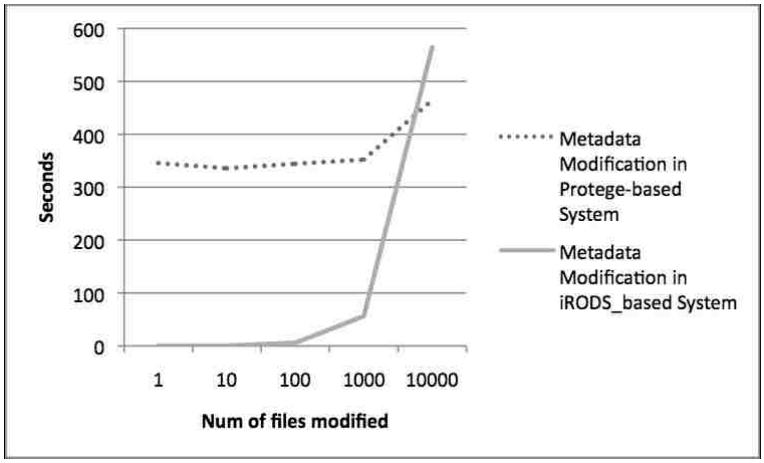

FIGURE 7.3: Performance Comparison of Metadata Modification

system has over Cross-Domain Metadata Management System in term of underlying implementation and metadata complexity. But as the amount of the files grew closer to 10000, performance gag between the two system rapidly got smaller, Cross-Domain Metadata Management System even outperformed iRODS system when size of experiment file set reached 10000 even though performance of Cross-Domain Metadata Management System followed a largely upwardly linear trend. We believe the performance discrepancy displayed by iRODS system can be explained by its lack of support for SQL or Sparql [14] style complex query languages, which meant iRODS system needs to modify metadata for one single file at a time. The overhead of repeated query to underlying relational database eventually outweighed the benefit generated by iRODS system's more efficient implementation and metadata simplicity.

7.4 Scalability Evaluation on Cross-Domain Metadata Management System

In previous section, we presented our work on benchmarking and comparing performances of Cross-Domain Metadata Management System in the context of datasets of limited size, namely, datasets with 10000 files were utilized to evaluate performances. In this section, we will present the work we have done to test the limit of scalability of Cross-Domain Metadata Management System in large scale data-intensive distributed environment. We will conduct our tests with a datasets containing 1 million dummy files.

As presented in previous section, Cross-Domain Metadata Management is based on ontology and Java. During our experiment, as we attempted to test the limit of the scalability of Cross-Domain Metadata Management System. The conflict of a Java-based system and the memory requirement for data-intensive applications was laid bare: Java Virtual Machine can only use at most 2 GB of memory in Linux system, which is hardly enough for a ontology containing metadata for tens of thousands of files. We experimented on inserting metadata for 1 million files in Cross-Domain Metadata Management System, the experiment ran 19 hours 12 minutes and 46.623 seconds, it succeeded in inserting metadata for 684632 files, then the process crashed when another Java-based program was launched. Another attempt ended with metadata for 734845 files inserted in 24 hours 43 minutes and 53.838 seconds, the process crashed again presumably because of memory hog. After changing the backend to relational database, we successfully tested insertion of 1 million instances on a workstation with 4 GB memory, Our experiment showed that 1 million instances could be inserted in 6898 minutes 59 seconds, approximately 5 days, sufficient to handle demands of scientific projects the system works with. It is also been observed that as the number of metadata grew, the execution of the insertion programs became extremely slow and unresponsive. It is clear that as the size of ontology grows, Cross-Domain Metadata Management System would encounter scalability problem but with proper backend data store, sufficient scalability could be achieved to satisfy demand.

Chapter 8

Data Object Discovery Performance and Scalability Evaluation[1]

8.1 Testbed

We implemented and tested our Data Object Metadata System on PetaShare, which is an state-level distributed data sharing cyber-infrastructure in Louisiana. It aims to enable collaborative data-intensive research in different application areas such as coastal and environmental modeling, geospatial analysis, bioinformatics, medical imaging, fluid dynamics, petroleum engineering, numerical relativity, and high energy physics. PetaShare manages the low-level distributed data handling issues, such as data migration, replication, data coherence, and metadata management, so that the domain scientists can focus on their own research and the content of the data rather than how to manage it.

Currently, there are seven PetaShare sites across Louisiana: Louisiana State University, University of New Orleans, University of Louisiana at Lafayette, Tulane University, Louisiana Tech University, Louisiana State University-Health Sciencies Center at New Orleans, and Louisiana State University-Shreveport. They are connected to each other via 40Gb/s optical network, called LONI (Louisiana Optical Network Initiative). In total, we have 300TB of disk storage and 400TB of tape storage on these sites. At each PetaShare site, we have an iRODS server deployed, which manages the data on that specific site.

Here, we use iRODS metadata system: iCAT, to store data object metadata. iRODS version in tested system is 2.2. iRODS is chosen because it is the decedent of SRB which was previously used in our group for data storage. Data Object Metadata System is replicated throughout the 7 PetaShare sites to avoid single point of failure at the expense of extra overhead on network bandwidth. The tests were done by piping all addition, deletion, etc. commands into imeta command.

[1]Reprinted by permission of “Scientific Programming”, published by IOS Press

TABLE 8.1: Testbed Sites Metrics

Testbed Sites Metrics				
Cluster	Peak Performance	# of nodes	Memory	Location
Eric	4.772 TFlops	128	4 GB/node	LSU
Oliver	4.772 TFlops	128	4 GB/node	ULL
Poseidon	4.772 TFlops	128	4 GB/node	UNO
Louie	4.772 TFlops	128	4 GB/node	Tulane

TABLE 8.2: Selective Data-Object Metadata

Selective Data-Object Metadata				
location	dateOfcreation	filetype	size	name
institution	creator	resolution	department	project

Table 8.1 illustrates metrics at some of the PetaShare sites. Table 8.2 provides a selective set of data-object metadata we use for performance and scalability benchmark testing, the set includes ten triples describing some of the data-object properties in our system.

8.2 Performance Benchmarking

Algorithm 1 Data-Object Metadata Insertion Performance Benchmarking Process

1: **while** BATCH INSERTION FILE NOT PROPERLY GENERATED **do**
2: recursively list all data objects need data object metadata annotation
3: formulate insertion commands for all data objects
4: output result to batch file
5: **end while**
6: execute batch file on data-object metadata store

For performance benchmarking, algorithms 1, 2, and 3 detail the process we employ to benchmark S_i, S_m, S_q respectively on testbeds described in the previous section. Figure 8.1, 8.2, 8.3 contain benchmarks of five rounds of performance tests on S_i, S_m, S_q respectively.

Each round of test consists of tests ranging in size from 1 to 10000 data objects, since metadata attached to each data object in our benchmarking tests consist of 10 triples, the maximum number of triples benchmarked in these tests is 100,000.

As illustrated in Figure 8.1, 8.2, as expected, performance of insertion and modification of data object metadata shows strong linear positive correlation to the number of triples involved.

Algorithm 2 Data-Object Metadata Modification Performance Benchmarking Process

1: formulate batch commands based on metadata triples need modification
2: execute batch file on distributed data sets

Algorithm 3 Data-Object Metadata Query Performance Benchmarking Process

1: **while** NOT SATISFIED WITH QUERY RESULT **do**
2: formulate query
3: execute query on data-object metadata store
4: return query in plain text
5: **end while**
6: formulate file access query
7: execute query on distributed data sets
8: acquire data object for further processing

The performance of insertion, improved considering the size of triples in data object metadata store increases ten times while the time taken to insert similar number of triples only doubles. Performance of modification, however, significantly deteriorates, even after considering the much more data intensive environment.

On the other hand, performance of query of data object metadata largely remains constant as the number of triples involved increases. In terms of absolute performance, however, query of data object metadata does not perform as well as hoped as time taken to finish a query that returns relatively small number of data objects still reaches several minutes, the relatively unsatisfactory performance of data object query is related to the size of the dataset, namely, dataset contains up to 1 million data objects and metadata store has up to 10 millions triples stored, in a less data intensive environment, performance of query operation should conceivably improve.

8.3 Scalability Benchmarking

Our attempts to scale our previous primarily ontology-based metadata schema to one million instances, each instance contains all the metadata pertinent to one individual data object in the system, failed dozens of times because of the extra overhead needed for accommodating ontology-based metadata schema in fine granularity. In this section, we mainly present our scalability benchmarking result conducted based on the new layered metadata management system illustrated in

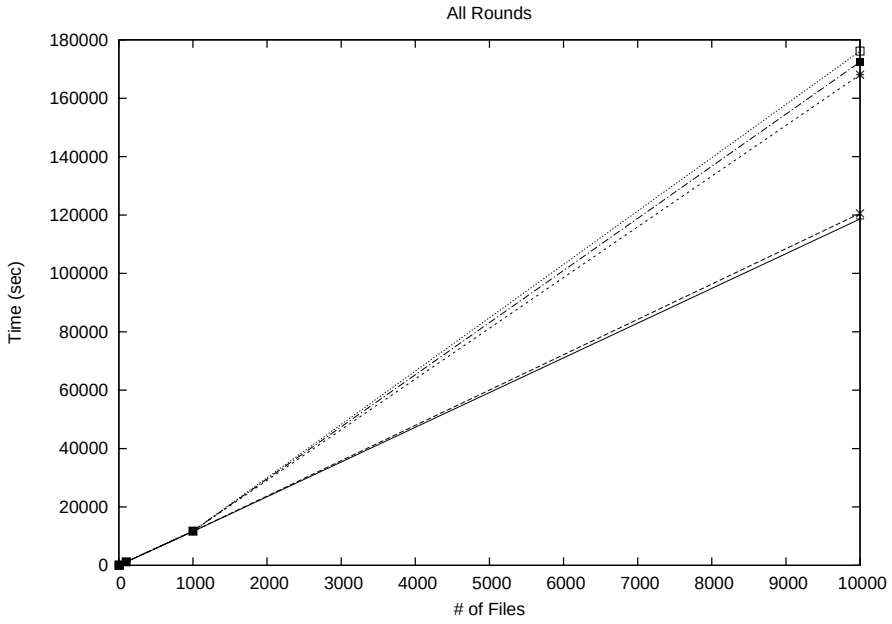

FIGURE 8.1: Data Object Insertion Performance Benchmarking

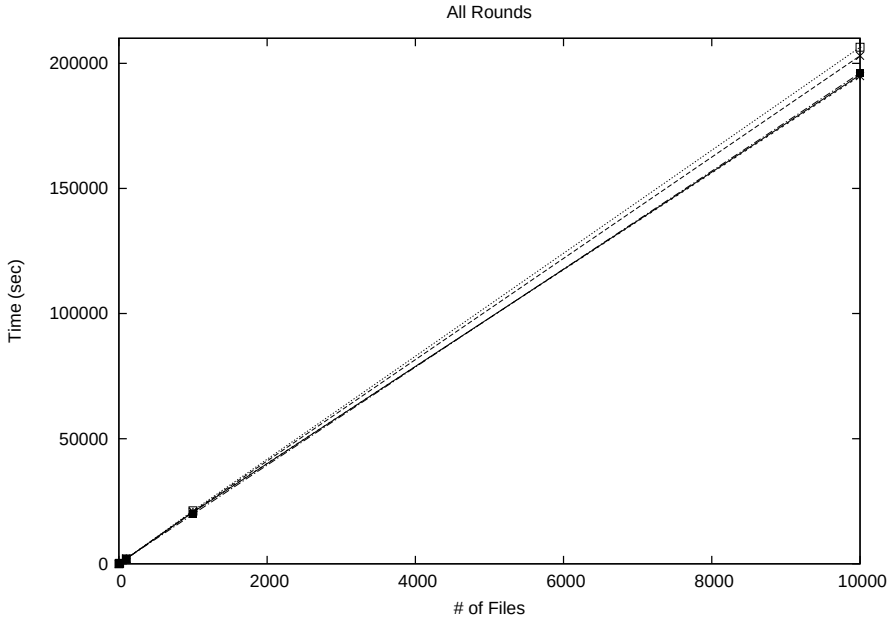

FIGURE 8.2: Modification Performance Benchmarking

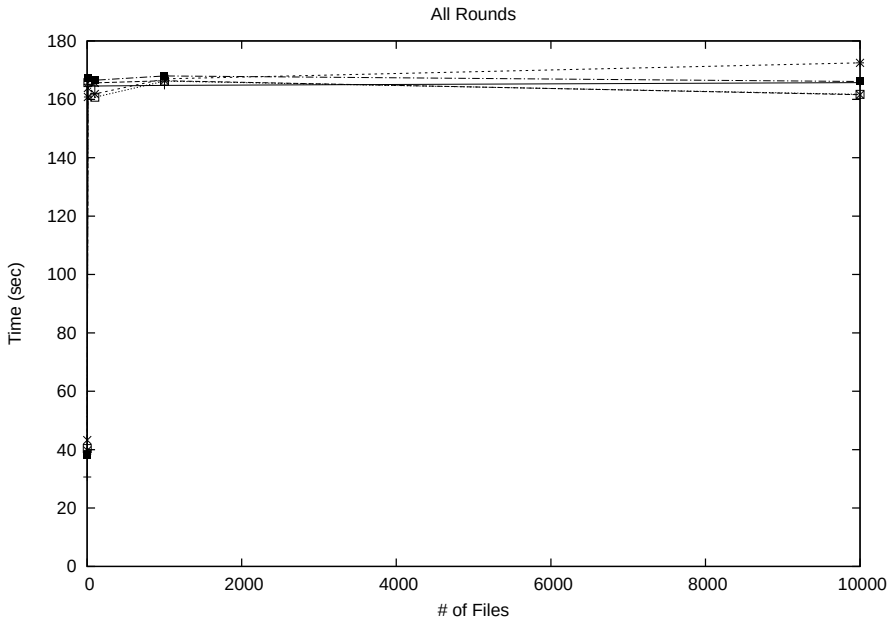

FIGURE 8.3: Query Performance Benchmarking

TABLE 8.3: S_o Before and After Each Round of Test

S_o Before and After Each Round of Test		
Round of Test	S_o before Test	S_o after Test
1	0	100,000
2	100,000	300,000
3	300,000	600,000
4	600,000	1,000,000

previous section. Specifically, we conduct tests on data-object metadata scalability S_o, as data-object metadata scalability are the primary piece of metadata information that every individual data object in the system needs to be annotated with, it is the biggest factor in degrading performance of metadata management system in a data-intensive computing environment. In our tests, we conducted four rounds of tests.

First round consists of attaching data object metadata to 100,000 data objects; second round consists of attaching data object metadata to 200,000 data objects; third and fourth rounds each consists of attaching data object metadata to 300,000 and 400,000 data objects respectively. As illustrated in Table 8.2, each individual data object will be attached with an set of ten triples data

object metadata. So together, after 4 rounds, the system contains ten millions triples in total. As a result, data-object metadata scalability S_o will be assigned a value $S_o = 10,000,000$, a vastly improved scalability benchmark than our previous experiment results. Number of S_o in the system before and after each round of test is listed in Table 8.3 .

Chapter 9
Semantically-Aware Data Placement

In this chapter, we describe different experimental scenarios and corresponding data placement strategies we employ to improve performance and throughput of data access by various related research groups.

9.1 Metadata Standards

In order to make our experimental scenarios as realistic as possible, we decide to take advantage of real-world cases in which semantically-aware data placement could improve performance and throughput. Below are brief introduction to the various data encoding standards, namely, NWIS from USGS and WQX from EPA, we seek to employ in our experimental scenarios as well as brief introduction of some of the current efforts to develop a standardized metadata vocabulary for hydrologic datasets.

In the US, Environmental Protection Agency (EPA), US Geological Survey (USGS) and National Oceanographic and Atmospheric Administration (NOAA) are the primary sources of water quality, quantity and climate datasets. While there are overlaps in data offerings, NOAA is the main source of meteorological data, USGS stands out with its extensive water quantity (surface/subsurface) data whereas EPA focuses on datasets on environmental quality. Heterogeneity is a major issue dealing with these datasets that are related yet tagged with different metadata standards.. USGS data is available, via the National Water Information System (NWIS) in different formats including delimited text, HTML tables and USGS own HydroML markup language. EPA is moving from delimited text to XML-based WQX (Water Quality eXchange) format. In addition to different encodings, there is no common vocabulary either. Lack of standards for hydrologic data exchange is a major problem a solution to which would eliminate the need for human involvement in data retrieval thus not only saves valuable research time but also makes it possible to implement

TABLE 9.1: NWIS Metadata and WQX Equivalents

NWIS Metadata and WQX Equivalence		
NWIS Parameter ID	Parameter Description	Equvalent WQX Charactername
00004	Stream width, feet	Instream features, est. stream width
00010	Temperature, water, degrees Celsius	Temperature, water
00011	Temperature, water, degrees Fahrenheit	Temperature, water, deg F
00020	Temperature, air, degrees Celsius	Temperature, air, deg C
00021	Temperature, air, degrees Fahrenheit	Temperature, air, deg F

automated workflows. This has been the main motivation behind the water data services part of the Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) Hydrologic Information Systems (HIS) project . The HIS project's experience in developing solutions to standardized access to hydrologic data sources in the United States demonstrates the challenges associated with establishing community semantics of hydrologic data exchange, formalizing the main notions of hydrologic observations, and evolution towards compliance with general data exchange protocols for cross-domain interoperability.

9.2 Experiment Scenario

This is the simplest scenario in which semantically-aware data placement strategies are employed to improve performance. In this scenario, we have one research group uploading observational sensor data encoded in NWIS metadata to data centers, as a result of the existence of equivalence relationships between NWIS standard and WQX standard, as partially described in Table 9.1, the system intelligently places the observational dataset among data centers it manages to achieve optimal performance/throughput for potential users of this dataset.

Below is description of testbed used for testing semantically-aware data placement strategies:

1. number of nodes in the system.

9 data storage nodes and one router node.

2. Node types and connectivity.

Node types:

Nodes belong to the same institutions.

Nodes belong to different institutions.

Router nodes.

Connectivity:

Nodes in the same institutions connected via 1Gbps network.

Nodes in the same institutions connected via 100 Mbps network.

Nodes in the same institutions are connected with Switches.

Nodes in different institutions are connected via switches and routers.

Routers connect different institutions via 100bps network or 10Mbps network.

3. Project

Project can be randomly distributed across the whole network, namely, project can contain nodes from the same or different institutions connected via fast or slow network.

Figure.9.1 illustrates the network topology used for test data placement strategies.

9.3 Semantically-Aware Data Placement Strategies

In any data placement/replication strategy, the most important factor impacting data placement strategy is the amount of available resource. Ideally, all relevant datasets should be placed on all disks to achieve the highest locality and best possible performance. However, as the size of datasets grows, it is clearly not realistic to implement the ideal strategy. This strategy would also incur the heaviest overhead since all datasets need to be placed on all servers, thus placing the heaviest workload on network whenever new data is generated. Assuming network throughput per second from node i to node j is S_{ij} and available disk space for data placement on node i is D_i, goal of data placement strategy would be to achieve $Min(\frac{D_{j1}}{S_{ij1}} + \frac{D_{j2}}{S_{ij2}} ..... + \frac{D_{jm}}{S_{ijm}})$, here node 1 to m denote m nodes chosen for placing data set relevant to groups working on node i. One strategy to achieve $Min(\frac{D_{j1}}{S_{ij1}} + \frac{D_{j2}}{S_{ij2}} ..... + \frac{D_{jm}}{S_{ijm}})$ would be to greedily choose node j to which node i has the highest network throughput and place the largest files in dataset until D_j is filled up.

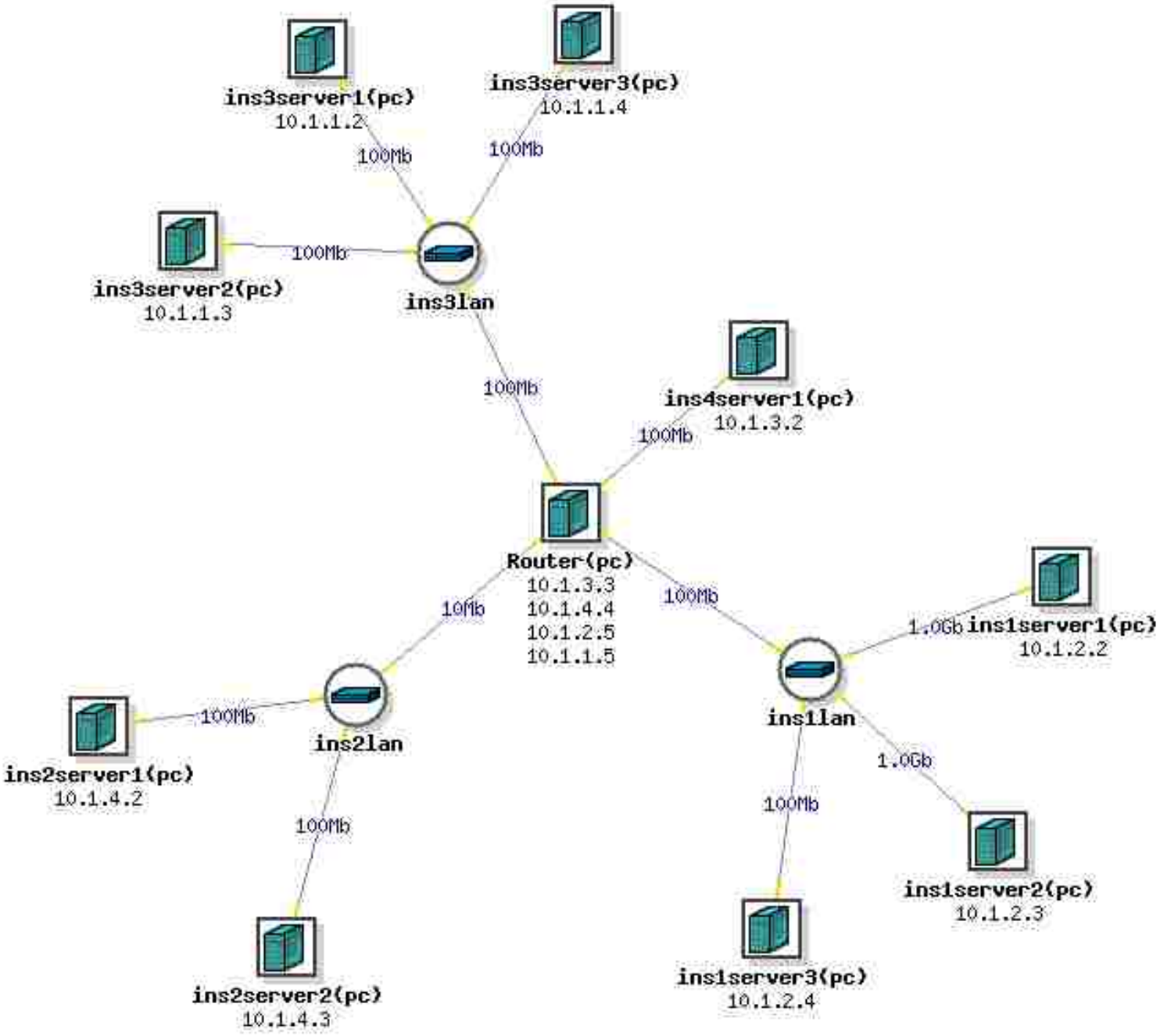

FIGURE 9.1: Network Topology

For example, assuming simulation data encoded in NWIS metadata standard has been produced and locally stored on node ins1server1 and simulation data encoded in WQX metadata standard

has been produced and locally stored in node ins3server2. By maintaining a periodically updated network throughput matrix, as shown in Table 9.2 and Figures 9.3, 9.4 and 9.5, the system

	ins1server1	ins1server2	ins1server3	ins2server1	ins2server2	ins3server1	ins3server2	ins3server3	ins4server1
ins1server1	local	26.31578947	4.173586696	0.886732014	0.846430774	0.741603778	0.736923885	0.776064331	0.753771778
ins1server2	23.91841778	local	5.628927158	0.885198655	0.889013448	0.745437805	0.813175545	0.739060109	0.745541201
ins1server3	5.783370195	5.859375	local	0.878084641	0.878563061	5.984782878	5.822175417	6.068303697	6.016416889
ins2server1	1.107467248	1.105480556	1.105764835	local	8.831986855	1.107682796	1.10675147	1.111762272	1.107841341
ins2server2	1.113194956	1.114303978	1.112203197	8.691551004	local	1.113111708	1.114387404	1.106371787	1.122064366
ins3server1	8.709938783	8.709938783	8.793855663	0.933636986	0.937654459	local	8.875739645	8.773918564	8.920750542
ins3server2	8.860701529	8.7890625	8.978285078	0.925775307	0.891774932	8.826749384	local	8.993725308	8.938470067
ins3server3	8.890011945	8.889195149	8.943427621	0.912137796	0.926764085	8.915407298	8.916639786	local	8.862324815
ins4server1	5.385621643	5.359516951	6.089309878	0.847457627	0.881665459	6.099098531	6.099098531	6.153797227	local

FIGURE 9.2: Network Throughput Matrix

would be able to place the semantically-related data set across available nodes to best reflect current hardware conditions (available resource, network throughput, etc.) so that relevant parties would be able to access these datasets in a optimal way, based on the network throughput matrix, the simplest strategy would be to distribute NWIS and WQX data sets to ins1server1, inst1server2, ins1server3 to achieve optimal performance since the 3 nodes chosen would present the best possible nodes, performance wise, for both ins1server1 and ins3server2, the original producers and users of NWIS and WQX datasets. On the other hand, the optimal nodes chosen for data placement need to have sufficiently large disk space to handle the data, in scientific simulation, datasets can come in two basic types:

1. Datasets with large individual files, such as visualization data set comprised of high-definition video files.

2. Datasets comprised of small individual files, such a water, soil and atmosphere time period observational files collected from tens of thousands of sensors spread across the nation.

Greedy strategy dictates that largest files should be placed on nodes with the best throughput to related project nodes, however, if datasets generated were comprised mainly of high-definition video files as often the case in scientific visualization, it is entirely possible that file size would exceed available space on chosen nodes. For example, in the above mentioned example, if ins1server1 only had 100 MB of available space, it would not be possible to place the largest file in the data base, which are of larger size. To deal with the problem, the system needs to gather information on available disk size on chosen nodes, then allocate files to nodes accordingly. Taken both throughput and available disk space into consideration, we have the following greedy algorithm that ensure $Min(\frac{D_{j1}}{S_{ij1}} + \frac{D_{j2}}{S_{ij2}} ..... + \frac{D_{jm}}{S_{ijm}})$. Issues need to be resolved before data could be placed include:

1. Which subset of nodes should be selected for data placement?
2. How to place different files to different nodes?
3. Is this strategy optimal?

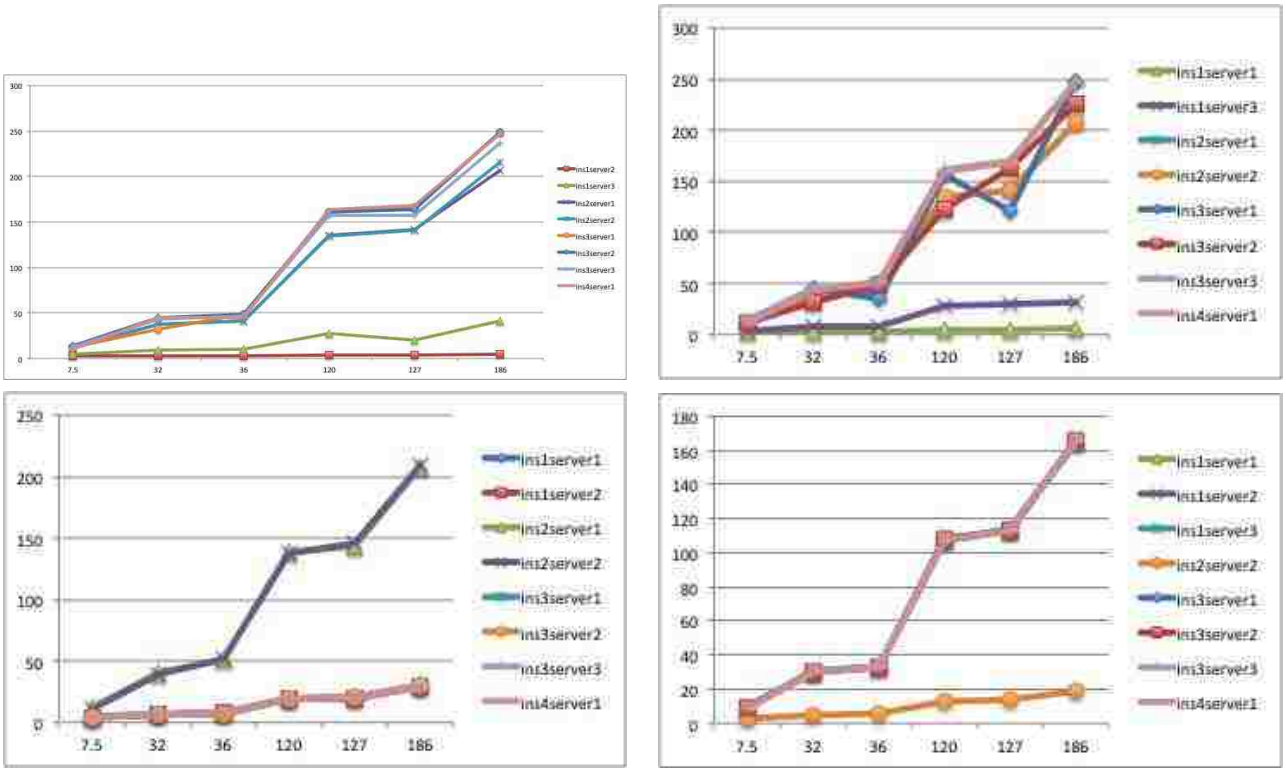

FIGURE 9.3: Access Performance

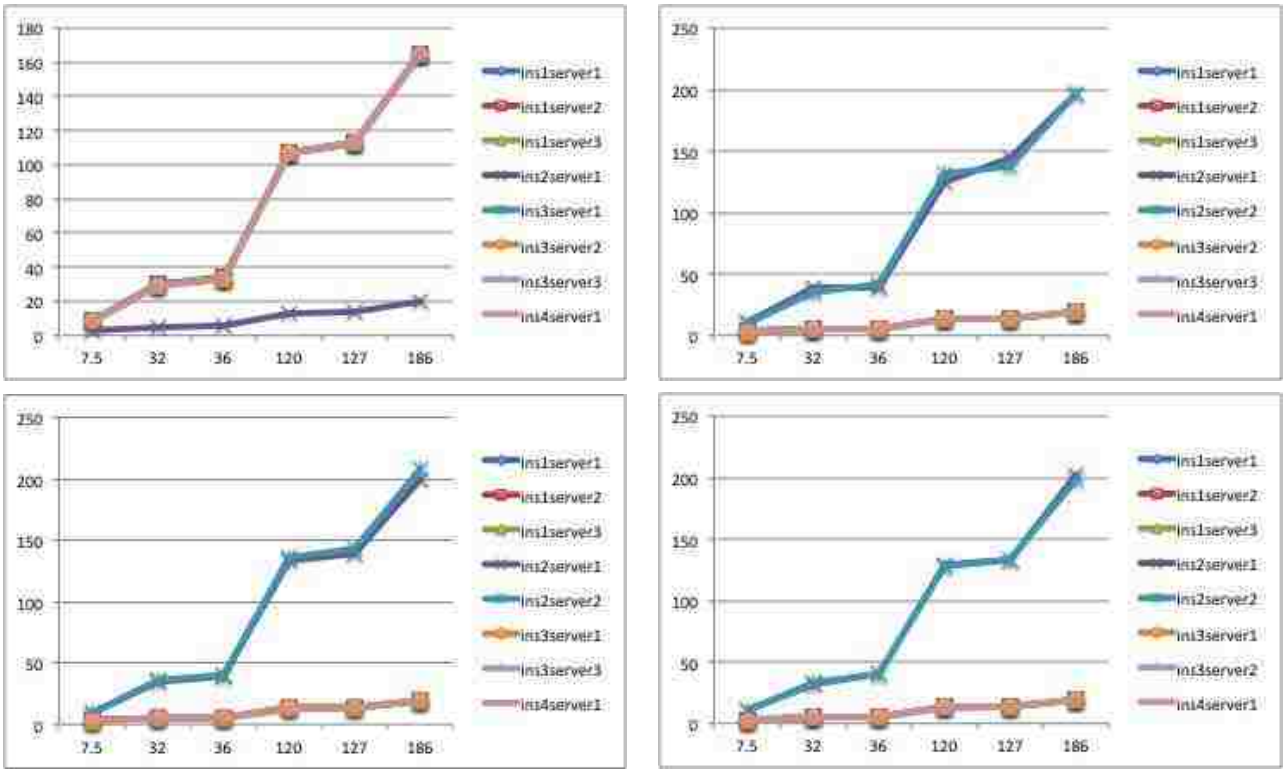

FIGURE 9.4: Access Performance (Continued)

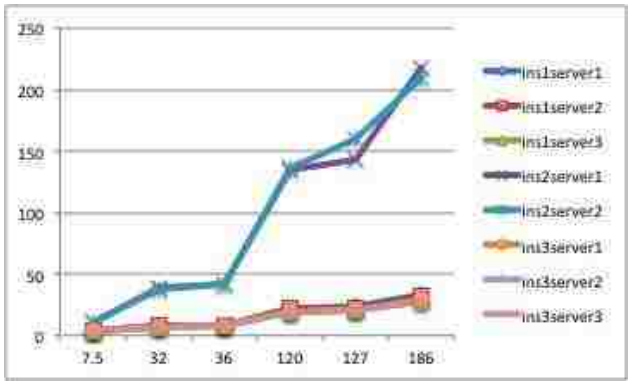

FIGURE 9.5: Access Performance (Continued)

For issue 1, first the system needs to weed out all nodes with available disk space smaller than the smallest files in dataset so that the system will not select a node that physically can not accept any file from the dataset; then the system needs to select a candidate set of nodes that could collectively store the dataset and achieve optimal performance for all relevant projects. Figures 9.6 shows nodes selected when size of candidate set of nodes is 1, 2, 3, 4, 5, 6, 7, 8, 9 respectively. In the

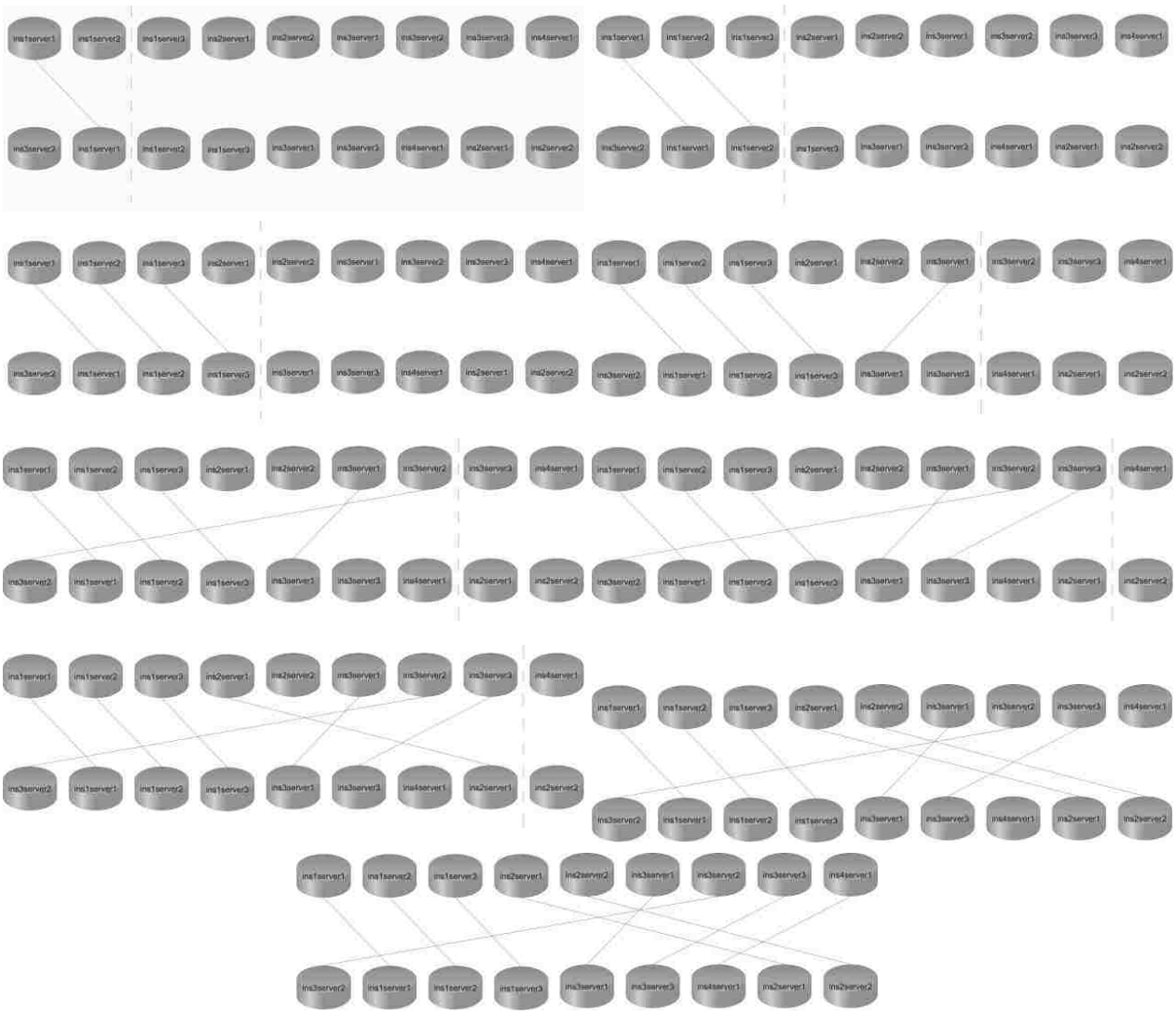

FIGURE 9.6: Candidate Selection

figures, each row of nodes is sorted in ascending order, to get candidate nodes, as shown in figures, is equivalent to solving two problems:

1. the candidate nodes collectively are sufficient for placing the whole data set.

2. the candidate nodes ensure optimal performance for relevant projects.

The first problem can be solved by only choosing minimal number of nodes needed for data placement, in another word, to achieve $Min(Sum(D_{j1} + D_{j2}..... + D_{jm}) - S_{dataset})$, here D_{j1} to D_{jm} are available space on respective nodes and $S_{dataset}$ is size of dataset. The second problem can be solved by algorithm 4:

Algorithm 4 Algorithm to Generate Candidate Nodes

```
1: create hash table T with node name as key and set all values in T to 0
2: select leftmost node as current
3: while current is smaller than size of node list do
4:    T[current] = T[current] + 1
5:    S_current = S_current - D_current
6:    if S_current > 0 and T[current] == 2] then
7:       add current node to candidate list
8:    end if
9:    current = current + 1
10: end while
```

The above algorithm only has to traverse through each row once to produce the candidate list that, assuming all relevant projects have equal probability of accessing the dataset, ensures optimal performance to relevant projects as a whole. Hence, the time complexity of algorithm 4 is O(n) with n denotes the number of nodes in the system.

After candidate nodes list is generated, algorithm 5 can be used to optimally place dataset to candidate nodes:

Algorithm 5 Greedy Data Placement Algorithm

```
1: gather available disk space information on nodes inside the system
2: create file list according to file size
3: sort file list in descending order
4: create nodes lists from all relevant rows in throughput matrix, only include nodes where avail-
   able disk space were larger than the smallest file
5: sort the nodes lists in descending order
6: create candidate nodes list out of nodes lists using algorithm 4
7: choose the first file from file list as current file
8: choose the first node from candidate nodes list as current node
9: while file list not empty do
10:    while available space on current node smaller than size of current file do
11:       if available space on current node smaller than size of current file then
12:          choose next node from candidate nodes list as current node
13:       else
14:          place current file to current node
15:          reduce available space on current node in candidate nodes list accordingly
16:       end if
17:       choose next file from the file list as current file
18:    end while
19:    choose leftmost available node from candidate nodes list as current node
20: end while
```

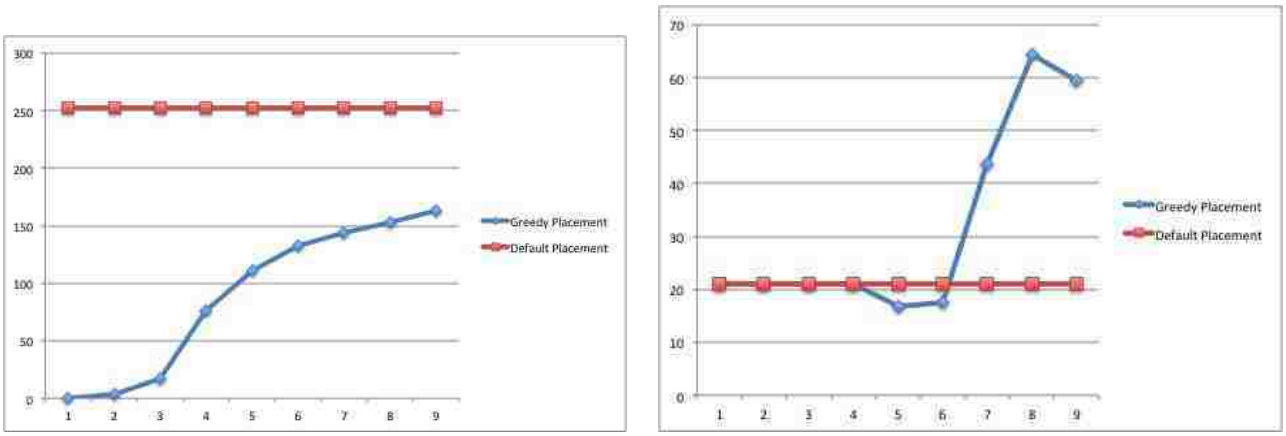

FIGURE 9.7: Performance Comparison Between Default Placement and Greedy Placement

Assuming there are n nodes in the system and m files in dataset, time complexity for step 1 would be O(n), for step 2 and step 3 will be O(m) + O(ln *m*) = O(ln *m*). Time complexities for step 4, 5 would be O(n) + O(ln *n*) = O(ln *n*), for step 6 would be O(n) as discussed above. Time complexity for step 7 to step 18 would be O(nm). Therefore, overall time complexity for algorithm 5 = O(n) + O(ln *m*) + O(ln *n*) + O(nM) = O(nm).

In Figure 9.7, expected performance comparisons of greedy placement discussed above and default placement, which automatically places all locally-produced on local nodes, are shown. Left side figure of Figure 9.7 shows expected performance comparison, depending on number of nodes greedy placement distributes data to, in the example discussed above, namely, when node ins1server1 trying to access data generated at node ins3server2, in the figure, red line indicates time required to access a file of size 186 MB stored on ins3server2 from ins1server1 and blue line indicates expected time required to access file of the same size placed by greedy strategy. As illustrated in the figure, as the number of nodes placed with data grows, the performance of acquiring file with a set size also deteriorates, clearly because addition of nodes with worse network throughput increases expected overhead. Still, greedy placement overall offers markedly better expected performance than default placement. In this case, it is obvious that greedy placement is a much better fit to place data locally stored on node ins3server2 for node ins1server1 to access.

On the other hand, the right half of Figure 9.7 shows that when it is node ins3server2 accessing data generated at node ins1server1, default placement offers better performance than the expected performance of greedy placement. Clearly, in this case, it is not worth the effort to apply greedy placement to data generated at node ins1server1, node ins3server2 could be better served by asking node ins1server directly for data.

The difference between the two scenarios illustrated in Figure 9.7 shows that even though it is desirable to apply greedy placement under certain circumstances, it is also likely that default placement offers better performance under difference circumstances. The factor determining choice of placement strategy here is network throughput between node generating the data and node that might need to access the data. As shown in network throughput matrix 9.2, in the scenario discussed above, there is significant difference between performances in the case of node ins1server1 accessing node ins3server2 and reverse. Node ins3server2 accessing node ins1server1 already enjoy near optimal network throughput, therefore, placing data on other nodes will not improve performance significantly, on the contrary, it will actually force node ins3server2 to access data placed on node with worse network throughput. Therefore, to help the users access node optimally, the system needs to make intelligent choice in choosing the data placement strategies.

Chapter 10
Conclusions

10.1 Contributions

In this dissertation, we seek to leverage semantically-aware metadata to enhance data discovery and placement to improve efficiency and performance of data management in collaborative computing environment. The dissertation will contribute to the aforementioned topics in the following ways:

1. Our system boosts scientists' ability to interoperate with each other by relieving them from the need to manually conduct the necessary mapping and "translation" required for accessing data archives covering multiple domains, while at the same time still affording scientists continued use of familiar terms and vocabularies. Scientists of one domain will be empowered to access data annotated and described by not only terms and vocabularies familiar to them, but also data from other domains annotated with vocabularies commonly used in respective domains. We achieve this goal by developing a metadata schema elastic enough to include metadata related to describing characteristics of scientific domains, metadata description of features of individual files/folder as well as conceptual and semantic mapping required for the integration of terminologies of different scientific domains.

2. Avoiding the pitfalls plaguing some of the previous systems, namely, the conflicts arise out of the need to be more descriptive in metadata development and the increasing burdensome overhead created by increasingly more descriptive metadata and increasingly larger datasets, our layered approach toward system implementation leads to lower footprint for metadata management in the overall data intensive collaborative computing infrastructure. We achieve this objective by implementing a metadata management system in a data-intensive collaborative distributed computing environment through a layered approach, we will separate high-level terms and vocabularies, which are needed to describe the domain as well as providing contextual information about data

object in its respective domain, from more technical metadata related to simulation and physical characteristics of actual data object such as files and folders, thus removing the need to attach high-level metadata not directly related to data objects which have become increasingly numerous as the size of data archive grows.

3. We conduct evaluation and benchmarking of the system we developed to test the efficiency and effectiveness of our implemented system, we describe a model with parameters most important to the overall quality of metadata management system so that future developer of similar system could be helped in the design process.

4. We also leverage semantically-aware metadata to help data placement become more intelligent. As data management in collaborative environment not only requires data to be discovered, it also requires data to be acquired in an efficient and cost-effective fashion. In this dissertation, we also propose and experiment with data placement algorithms that leverage semantically-aware information that include hardware semantics like available disk space and network traffic, but also semantic metadata that could increase semantic locality. As our literature review indicates, so far, none of the current data placement strategies leverage such information.

5. The work also contributes to system engineering addressing data management problems in the context of scientific computing, which presents unique challenges that current available commercial and open-source solutions do not tackle sufficiently.

To sum it up, this dissertation describes important challenges facing the management of large-scale datasets in collaborative computing environment in scientific computing, namely, the issue of discovering data relevant to users as well as the issue of placing data so that users could access efficiently, examines existing works done to address these issues, presents and discusses our novel approaches and conducts experiments to test performance and scalability of proposed solutions.

10.2 Future Works

As presented in this dissertation, the system presented and the problems the system attempts to address are distributed and scalable, but the scalability achieved is far from what a truly national

or even international cyber-infrastructure for science need to possess. The compromise and design decisions made as part of this work have not been, due to the limitation of resources and datasets, been tested on an scale on par with solutions provided by industry to problems of similar nature. Therefore, the most important part of future works should be to scale up the system, both in term of hardware and dataset, but also regarding number of disciplines and metadata standards, in another word, both system and semantic scalability need to be tested in a much bigger scale, as the system scales up, the system also needs to consider the threat of node failures and how best to build sufficient redundancy into the system so that failures would not fatally affect availability and redundant resources are not seriously underutilized. Also, this work does not touch upon the problem of data processing, integration of data processing capabilities into the system is critical to the construction of a full-spectrum data management system. On the theoretical side, the task of modeling such a highly distributed system, in term of performance, scalability, and extensibility is a very hard problem, which is also a hot research topic and necessary part of the future works if commonalities were to be extracted for devising solutions to solutions to similar problems in the future. Specifically, the future works are needed:

1. Increase the number of and heterogeneity of projects and metadata standards involved in tests

2. Increase size of, number of and level of distribution of datasets involved in tests.

3. Using new test data to reexamine the compromises and design decisions.

4. Integration of failure-handling mechanism to handle constant node failures, which are inevitable in a large-scale distributed system.

5. Integration of existing data processing systems such as Apache Hadoop[24].

6. Building a continuous monitoring module to monitor the system with the hope that with enough historical data, a more concrete and stable model could be built to assess the quality of solutions to similar problems.

Bibliography

[1] A Semantic Web Framework for Java. http://jena.sourceforge.net/.

[2] Amazon simpledb. http://aws.amazon.com/simpledb/.

[3] Apache hbase. http://hbase.apache.org/.

[4] Cuahsi hydrologic information system. http://his.cuahsi.org/.

[5] DAML-ONT Initial Release. http://www.daml.org/2000/10/daml-ont.html.

[6] Dublin core. http://dublincore.org.

[7] Earth science curator. http://www.earthsystemcurator.org/.

[8] Earth science modeling framework. http://www.esmf.ucar.edu/.

[9] KAON - The KArlsruhe ONtology and Semantic Web Tool Suite. http://kaon.semanticweb.org/.

[10] The numerical relativity group. http://www.cct.lsu.edu/numerical/index.php.

[11] Owl web ontology language overview. http://www.w3.org/TR/owl-features/.

[12] Protege ontology editor. http://protege.stanford.edu/.

[13] Simple HTML Ontology Extensions. http://www.cs.umd.edu/projects/plus/SHOE/.

[14] Sparql query language for rdf. www.w3.org/TR/rdf-sparql-query/.

[15] The sura coastal ocean observing and prediction (scoop). http://scoop.sura.org/.

[16] Ubiquitous computing and monitoring system. http://www.ucoms.org/.

[17] Pinar Alper, Oscar Corcho, and Carole Goble. Understanding semantic aware grid middleware for e-science. *Computing and Informatics*, 27:93–118, 2008.

[18] Atlas. European Organization for Nuclear Research(CERN) A Toroidal LHC ApparatuS(ATLAS) Project. http://atlasexperiment.org/, 2008.

[19] Mehmet Balman, Ibrahim Suslu, and Tevfik Kosar. Distributed data management with petashare. In *MG '08: Proceedings of the 15th ACM Mardi Gras conference*, pages 1–1, New York, NY, USA, 2008. ACM.

[20] C. Baru, R. Moore A., Rajasekar, and M. Wan. The SDSC Storage Resource Broker. In *Proceedings of CASCON*, Toronto, Canada, 1998.

[21] D. Calvanese, G.D. Giacomo, and M. Lenzerini. Description logics for information integration. *Computational Logic: From Logic Programming into the Future, Springer–Verlag*, 2001.

[22] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.*, 26(2), 2008.

[23] CMS. PPDG Deliverables to CMS. http://www.ppdg.net/archives/ppdg/2001/doc00017.doc, 2008.

[24] Apache Foundation. Apache hadoop. http://hadoop.apache.org/.

[25] Joe Futrelle. Developing metadata standards for scientific data reuse in ncsa's distributed grid architecture. Technical report, National Center for Supercomputing Applications, 2006.

[26] Lei Gao. Sar: Semantic-aware replication, 2005.

[27] Y. Gil, V.Ratnakar, and E.Deelman. Metadata Catalogs with Semantic Representations. In *Proceedings of IPAWS*, 2006.

[28] T. Goodale, G. Allen, G. Lanfermann, J. Massó, T. Radke, E. Seidel, and J. Shalf. The Cactus Framework and Toolkit: Design and Applications. In *Vector and Parallel Processing - VECPAR '2002, 5th International Conference*. Springer, 2003.

[29] GOP. The Gene Ontology Project. http://www.geneontology.org/, 2008.

[30] T. R. Gruber. A Translation Approach to Portable Ontologies. *Knowledge Acquisition*, pages 199–220, 1993.

[31] Yu Hua, Hong Jiang, Yifeng Zhu, Dan Feng, and Lei Tian. Smartstore: a new metadata organization paradigm with semantic-awareness for next-generation file systems. In *SC*, 2009.

[32] S. J. Jeffrey and J. Hunter. A Semantic Search Engine for the SRB, 2006.

[33] Z. Kaoudi, M. Koubarakis, K. Kyzirakos, M. Magiridou, I. Miliaraki, and A. Papadakis-Pesaresi. Publishing, discovering and updating semantic grid resources using dhts. In *In Proc. of the CoreGRID Workshop on Grid Programming Model, Grid and P2P Systems Architecture, Grid Systems, Tools and Environments*, 2007.

[34] Vipul Kashyap and Amit Sheth. Semantic heterogeneity in global information systems: The role of metadata, context and ontologies. In *Cooperative Information Systems: Current Trends and Directions*, pages 139–178. Academic Press, 1998.

[35] Tevfik Kosar and Miron Livny. Stork: Making data placement a first class citizen in the grid. In *ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pages 342–349, Washington, DC, USA, 2004. IEEE Computer Society.

[36] Luis E. Bermudez. ONTOMET: Ontology Metadata Framework. http://hdl.handle.net/1860/376f, 2004.

[37] J. MacLaren, G. Allen, C. Dekate, D. Huang, A. Hutanu, and C. Zhang. Shelter from the storm: Building a safe archive in a hostile world. In *On the Move to Meaningful Internet Systems*, 2005.

[38] A. Maedche, B. Motik, and L. Stojanovic. Managing multiple and distributed ontologies on the semantic web. *VLDB Journal*, 12:286–302, 2003.

[39] Boris Motik, Raphael Volz, and Sean Bechhofer. DLP: Description Logic Programs. http://kaon.semanticweb.org/alphaworld/dlp, 2007.

[40] Zhengxiang Pan, Abir Qasem, and Jeff Heflin. An investigation into the feasibility of the semantic web. In *In Proc. of the Twenty First National Conference on Artificial Intelligence (AAAI 2006*, pages 1394–1399, 2006.

[41] Kaustubh Supekar Quddus Chong, Anup Marwadi and Yuyung Lee. Ontology Based Metadata Management in Medical Domains. *Journal of Research and Practice in Information Technology*, pages 139–154, 2003.

[42] SDSC. San Diego Supercomputer Center iRODS project. https://www.irods.org/, 2008.

[43] John Dunagan Sharad Agarwal et al. Volley: Automated data placement for geo-distributed cloud services. In *NSDI*, 2010.

[44] Gurmeet Singh, Shishir Bharathi, Ann L. Chervenak, Ewa Deelman, Carl Kesselman, Mary Manohar, Sonal Patil, and Laura Pearlman. A metadata catalog service for data intensive applications. In *SC*, page 33, 2003.

[45] Muthian Sivathanu, Lakshmi N. Bairavasundaram, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. Database-aware semantically-smart storage. In *FAST*, 2005.

[46] Muthian Sivathanu, Vijayan Prabhakaran, Florentina I. Popovici, Timothy E. Denehy, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. Semantically-smart disk systems. In *FAST*, 2003.

[47] H. Stuckenschmidt, H. Wache, T. Ogele, and U. Visser. Enabling technologies for interoperability. In *Workshop on the 14th International Sysposium of Computer Science for Environmental Protection*, Bonn, Germany, 2000.

[48] SWRL. World Wide Web Consortium. SWRL: A Semantic Web Rule Language Combing OWL and RuleML. http://www.w3.org/Submission/SWRL/, 2008.

[49] Hong Tang, Aziz Gulbeden, Jingyu Zhou, William Strathearn, Tao Yang, and Lingkun Chu. A self-organizing storage cluster for parallel data-intensive applications. In *SC '04: Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, page 52, Washington, DC, USA, 2004. IEEE Computer Society.

[50] James Hendler Tim Berners-Lee and Ora Lassila. The Semantic Web. *Scientific American*, 2001.

[51] Xinqi Wang and Tevfik Kosar. Design and implementation of metadata system in petashare. In *SSDBM*, pages 191–199, 2009.

[52] Zhichen Xu, Magnus Karlsson, Chunqiang Tang, and Christos T. Karamanolis. Towards a semantic-aware file store. In *HotOS*, pages 181–187, 2003.

Appendix A: Inderscience Reprint Permission

AUTHOR AGREEMENT: EXPLANATORY NOTES

Inderscience's policy is to acquire copyright for all contributions, for the following reasons:

a ownership of copyright by a central body helps to ensure maximum international protection against infringement and/or plagiarism;

b requests for permission to reproduce articles in books, course packs, electronic reserve or for library loan can be handled centrally, relieving authors of a time-consuming administrative burden.

c the demand for research literature in electronic form can be met efficiently, with proper safeguards for authors, editors and journal owners.

There are opportunities to reach institutions (e.g., companies, schools and public libraries) and individual readers that are unlikely to subscribe to the printed journal. Inderscience works with other organisations to publish its journals online, and to deliver copies of individual articles. It has registered the Journal with the Copyright Licensing Agency, which offers centralised licensing arrangements for digital copying and photocopying around the world. Income received from all of these sources is used to further the interests of the Journal.

Once accepted for publication, your Article will be published in the Journal, and will be stored and distributed electronically, in order to meet increasing library and faculty demand and to deliver it as part of the Journal, as an individual article or as part of a larger collection of articles to meet the specific requirements of a particular market. By signing this Author Agreement and assigning copyright you agree to Inderscience making such arrangements.

It may be that the Author is not able to make the assignment solely by him- or herself:

a If it is appropriate, the Author's employer may sign this agreement. The employer may reserve the right to use the Article for internal or promotional purposes (by indicating on this agreement) and reserve all rights other than copyright.

b If the Author is a UK Government employee, the Government will grant a non-exclusive licence to publish the Article in the Journal in any medium or form provided that Crown Copyright and user rights (including patent rights) are reserved. This also applies to other Commonwealth countries.

c If the Author is a US Government employee and the work was done in that capacity, the assignment applies only to the extent allowed by US law.

Under the UK's Copyright Design and Patents Act 1988, the Author has the moral right to be identified as the author wherever the Article is published, and to object to its derogatory treatment or distortion. Inderscience encourages assertion of this right; it is represents best publishing practice and is an important safeguard for all authors. Clause 4 asserts the Author's moral rights, as required by the Act.

The Journal will permit the Author to use the Article for non-commercial purposes after publication:

- posting the Accepted Version (i.e. final post-acceptance manuscript version) on the author's personal web pages or in an institutional repository maintained by the institution to which the Author is affiliated, provided acknowledgement is given to the Journal as the original source of publication and upon condition that it shall not be accessible until after six months from Inderscience's publication date. For any queries about copyright, please contact copyright@inderscience.com.

- using the Article in further research and in courses that the Author is teaching, provided acknowledgement is given to the Journal as the original source of publication;

- incorporating the Article content in other works by the Author, provided acknowledgement is given to the Journal as the original source of publication.

Inderscience, as publisher, reserves the right to refuse to publish your Article where its publication creates legal liability, or where circumstances come to light that were not known to the Editor, including prior publication, conflict of interest, manifest error etc. Inderscience is the ultimate custodian of academic quality and integrity, and will ensure that this will be done only in exceptional circumstances and on reasonable grounds. In such circumstances the Article will be returned to the Author together with all rights in it.

Thank you for reading these notes. If you require more detailed information, go to Inderscience's web site. This assignment will enable Inderscience to ensure that the Article will reach the optimum readership.

Inderscience Enterprises Ltd, trading as Inderscience Publishers, of World Trade Center Building II, 29 Route de Pre-Bois, Case Postale 856, CH-1215 Geneva 15, Switzerland ("Inderscience")

Appendix B: IOS Press Reprint Permission

License to Publish

In order to publish your article we need your agreement. Please take a moment to read the terms of this license.

By submitting your article to one of our books or journals (henceforth 'publications'), you and all co-authors of your submission agree to the terms of this license. You do not need to fill out a copyright form for confirmation.

By submitting your article to one of our publications you grant us (the publisher) the exclusive right to both reproduce and/or distribute your article (including the abstract) throughout the world in electronic, printed or any other medium, and to authorize others (including Reproduction Rights Organizations such as the Copyright Licensing Agency and the Copyright Clearance Center, and other document distributors) to do the same. You agree that we may publish your article, and that we may sell or distribute it, on its own, or with other related material.

By submitting your article for publication to one of our publications, you promise that the article is your original work, has not previously been published, and is not currently under consideration by another publication. You also promise that the article does not, to the best of your knowledge, contain anything that is libelous, illegal or infringes anyone's copyright or other rights. If the article contains material that is someone else's copyright, you promise that you have obtained the unrestricted permission of the copyright owner to use the material and that the material is clearly identified and acknowledged in the text.

We promise that we will respect your rights as the author(s). That is, we will make sure that your name(s) is/are always clearly associated with the article and, while you do allow us to make necessary editorial changes, we will not make any substantial alterations to your article without consulting you.

Copyright remains yours, and we will acknowledge this in the copyright line that appears on your article. You also retain the right to use your own article (provided you acknowledge the published original in standard bibliographic citation form) in the following ways, as long as you do not sell it in ways that would conflict directly with our efforts to disseminate it.

a You are free to use the manuscript version of your article for internal, educational or other purposes of your own institution or company;

b You may use the article, in whole or in part, as the basis for your own further publications or spoken presentations;

c For a fee of €100, you will have the right to mount the final version of your article as published by IOS Press on your own, your institution's, company's or funding agency's website. You can order this right together with the final published version of your article with the form sent to the corresponding author along with the proofs of your paper.

Vita

The author, Xinqi Wang was born in Taiyuan, China on May 10th, 1979. The author enrolled in Taiyuan University of Technology in 1998 and received a Bachelor of Engineering degree in Computer Science in 2002. After graduating from college, he worked at Industrial and Commercial Bank of China for 6 month.

The author enrolled into the graduate program at Department of Computer Science of Taiyuan University of Technology in 2004, his area of research was semantic web, ontology modeling and artificial intelligence. He fulfilled all the course requirements, published 3 papers and was working on his master thesis while being employed as a research assistant between 2004 and 2007.

The author enrolled into the doctoral program at Department of Computer Science of Louisiana State University beginning January, 2007. He has been working as a graduate research assistant under Dr.Tevfik Kosar's supervision at Center for Computation&Technology, Louisiana State University between 2007 and 2011. His area of research is semantically-aware data discovery and placement in collaborative environment. He is also interested in high performance and distributed computing, data management and distributed algorithm.

The author worked as an summer intern at National Center for Atmospheric Research (NCAR) in Boulder, Colorado, under the supervision of Mrs.Sylvia Murphy and Mrs.Cecelia DeLuca of the Earth System Modeling Framework. His work at NCAR centered around developing APIs to enable cross-accessing of datasets produced by ESMF as well as Hydrologic Information System (HIS). He successfully finished his project and publicly defended it before NCAR researchers and students.

The author also spent one semester at Amazon Web Services LLC, in Seattle, Washington, working at Amazon Elastic Compute Cloud (EC2) division as the main developer for a new cloud-based service responsible for designing continuous cloud-based tests, scheduling these tests, collecting and analyzing test datas to make actionable decisions addressing system anomalies with the ulti-

mate goal of providing monitoring and quality assurance services to all of Amazon Web Services. He is involved in all aspect of the project and responsible for the development of the majority of the available features, such as monitoring, scheduling, data collecting and analyzing of tests on EC2 network latency and Elastic Block Store (EBS).

He has contributed 1 book chapter, 2 journal papers and 6 conference papers during his doctoral study, he also presented his works on 4 conferences.

He also served as reviewer for multiple international conferences.

His most significant publications are:

- Ismail Akturk, **Xinqi Wang** and Tevfik Kosar. *Distributed Storage Management with PetaShare*, in Data Intensive Distributed Computing: Challenges and Solutions for Large-scale Information Management, Editor: Tevfik Kosar. Publisher: IGI Gobal. DOI: 10.4018/978-1-61520-971-2, ISBN13: 9781615209712, ISBN10: 1615209719, EISBN13: 9781615209729.

- T. Kosar, I. Akturk, M. Balman, **X. Wang**. *PetaShare: A Reliable, Efficient, and Transparent Distributed Storage Management System*, Scientific Programming Journal, Volume 19 (1), pp.27-43, 2011.

- **X.Wang**, D. Huang, I. Akturk, M. Balman, G. Allen, T. Kosar. *Semantic Enabled Metadata Management in Petashare*, In International Journal of Grid and Utility Computing 2009 - Vol. 1, No.4 pp. 275 - 286.

- Ismail Akturk, **Xinqi Wang,** Tevfik Kosar. *Toward a Reliable Distributed Data Management System*, ISPDC 2010: 109-116

- **Xinqi Wang**, Ismail Akturk, Tevfik Kosar. *Cross-Domain Metadata Management in Data Intensive Distributed Computing Environment*, In Proceedings of 11th IEEE International Conference on Cluster Computing (Cluster2009), New Orleans, LA, USA, 2009

- **Xinqi Wang**, Tevfik Kosar. *Design and Implementation of Metadata System in PetaShare*, In Proceedings of 21st International Conference on Scientific and Statistical Database Management (SSDBM2009), New Orleans, LA, USA, June 2-4, 2009, 191-199.

- Dayong Huang, **Xinqi Wang**, Gabrielle Allen, Tevfik Kosar. *Semantic Enabled Metadata Framework for Data Grids*, In Proceedings of International Workshop on P2p, Parallel, Grid and Internet Computing (3PGIC-2008), Barcelona, Spain, March 2008

- **Xinqi Wang**, Ismail Akturk and Tevfik Kosar. *Cross-Domain Metadata Management in Cybertools*, In Proceedings of NSF EPSCoR Research Infrastructure Improvement (RII) Award: Cyberinfrastructure and Science Driver Symposium. Baton Rouge, LA, 2009.

- Ismail Akturk, Mehmet Balman, **Xinqi Wang**, Tevfik Kosar, Tyler Barker, Erik Schnetter, Rajuottumukkala. *Distributed Data Sharing with PetaShare for Collaborative Research in Cybertools*, In Proceedings of NSF EPSCoR Research Infrastructure Improvement (RII) Award: Cyberinfrastructure and Science Driver Symposium. Baton Rouge, LA, 2009.

- **Xinqi Wang**, Xueli Yu. *A OWL-Based Semantic Web Service Discovery Framework (Extended Version)*, In Proceedings of International Conference on Internet and Web Applications and Services (ICIW 2006). Guadeloupe, French Caribbean. February 19-25, 2006, 125.

- **Xinqi Wang**. *Towards a Situation-Based Service Composition Framework* In Proceedings of International Workshop on WWW Service Composition with Semantic Web Services (wscomp05), Compiegne, France, September, 2005, page 56-57.

- **Xinqi Wang**, Xueli Yu. *A OWL-Based Semantic Web Service Discovery Framework*, In Proceedings of International Workshop on OWL: Experiences and Directions 2005 workshop (owled2005), Galway, Ireland, November 11-12, 2005.

- Li Wang, Yingjie Li, Wen Li, **Xinqi Wang**, Yu Xing, Xueli Yu. *The Semantic Matching of Semantic Web Services*, In Proceedings of International Workshop on Knowledge Grid and Grid Intelligence (KGGI2004), Beijing, China, September 20, 2004.