

Design of a Peer-to-Peer Information System Using the GT4 Index Service

Shishir Bharathi, Ann Chervenak

USC-Information Sciences Institute

Marina del Rey, California, USA

shishir@isi.edu

annc@isi.edu

I. INTRODUCTION

Applications running on the Grid need efficient access to information about resources and services. This information can significantly impact scheduling, replica selection, and planning for workflow execution. Grid Information Services need to be able to aggregate information from multiple types of resources and support both explicit and attribute-based queries. As Grids grow larger, organizing these services into efficient hierarchies, while taking care to avoid cycles and ensure that nodes do not become hot spots, is challenging.

We present a peer-to-peer organization of a distributed Grid information service, the GT4 Index Service. In our design, modified GT4 Indexes called P2P Indexes organize themselves into an unstructured P2P system. Queries from a user to any index may be forwarded to other indexes, and query results are routed back to the user. We use the P2P network only for self-organization and query forwarding among P2P indexes, which are optimized to store resource information and process associated queries.

Most previous work in unstructured P2P networks has been on file sharing applications or data storage systems. Here, we explore the issues of organizing information services into an unstructured P2P overlay. Our goal is to provide a distributed service for Grid resource discovery that supports the features of a specialized information service such as the GT4 Index service but is also highly scalable and fault tolerant with self-organizing and self-healing properties of P2P systems.

II. DESIGN AND IMPLEMENTATION

The Globus Toolkit Version 4 Index Service is part of the Monitoring and Discovery System (MDS), a suite of Web Services-Resource Framework [1, 3] based services that provide monitoring and discovery of Grid services. The GT4 Index Service aggregates information from other Grid services and publishes it as a single Resource Property (RP) document. Queries to the GT4 Index Service consist of a query language identifier and a query expression. The default query processing engine provided with the toolkit processes XPath queries and returns matching XML content.

In our implementation, the P2P Index Service operates on a stateful P2PIndexServiceResource that is composed of an IndexServiceResource and a P2PResource, as illustrated in Figure 1. The IndexServiceResource component is responsible for storing information updated via standard MDS

mechanisms and also for query processing. The P2PResource component is responsible for creating and maintaining the P2P overlay and for forwarding and routing messages. The overlay is not used to replicate resource information among peers.

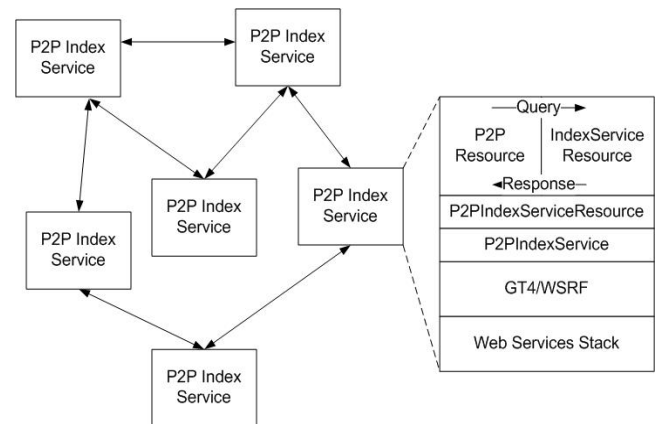


Figure 1 The P2P Index Service

The P2P Indexes in the Grid organize themselves into an unstructured overlay using an approach similar to those used in Gnutella-like systems. We use a flooding algorithm for message forwarding. Upon receiving a request message from a peer, a node processes the message and if required, forwards it to all its other peers. Responses are routed back along the path taken by the corresponding request message.

A “query” message is initially processed by the P2PResource component, which extracts the query part of the message and passes it on to the IndexServiceResource for processing. The results are bundled into a “query response” message that is routed back appropriately.

We chose to use an unstructured overlay network for our system because unstructured networks are easy to build and can support optimizations such as higher connectivity between certain nodes and locality-based modifications to the topology. We did not choose a structured overlay network since policy restrictions may prevent nodes from peering with or responding to queries from certain nodes.

Using a flooding algorithm in an unstructured overlay results in exponential growth in the number of messages sent in the network. A *query caching* scheme similar to the learning-based strategy described by Iamnitchi et al. [4] would forward a query only to peers that responded to that query in

the past. This reduces the number of messages sent in the network. We have implemented a query caching and probabilistic forwarding approach. Each node deterministically forwards a query to peers that responded to the same query earlier and probabilistically forwards the query to other peers that did not respond to the query earlier. This approach helps in identifying nodes that have been recently updated and in ensuring that caches are set up correctly.

III. EXPERIMENTS

We now present results of experiments conducted on a prototype implementation of the P2P Index Service. These tests were conducted with P2P Indexes deployed over GT4.0.1 on cluster nodes with dual 2.4GHz Xeon processors and 2GB memory running GNU/Linux from a Debian 3.1 distribution. The client system, also running Debian 3.1, had dual 2.2GHz Xeon processors and 1GB of memory.

We performed queries on the P2P network using Pegasus [2], a planning application that maps scientific workflows onto the Grid. We populated P2P Index service nodes with information from Pegasus site catalogs and transformation catalogs. We also integrated a query module into Pegasus that queries for this information over the P2P network. The information uploaded into the GT4 Index and the P2P Indexes in our tests is similar in content to that provided by real world Grid sites that use Pegasus for planning workflow execution.

Figure 2 shows a comparison between the times taken by Pegasus to generate a plan for a simple 100 node workflow when querying a single unmodified GT4 Index and a single P2P Index. The performance of both indexes depends mainly on the performance of the XPath query processing engine (built on top of Apache xalan-j) used to process the XPath queries from clients. We can see that the overhead of the P2P framework is relatively low and fairly constant as we increase the amount of information stored in the index.

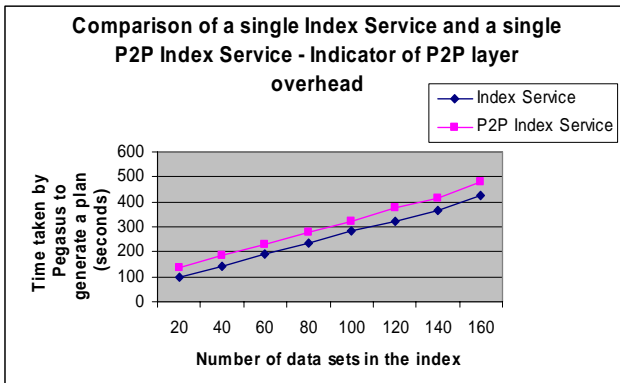


Figure 2 Comparison of the GT4 Index Service and the P2P Index Service

Next, we distributed resource information regarding 160 sites uniformly over 8 index service nodes in the cluster. We varied the number of peers each node had from 2 to 7 and measured the time taken by Pegasus to generate a plan for the 100 node workflow in each configuration. We enabled caching and probabilistic forwarding at each node. Setting the forwarding probability to 0 is equivalent to enabling a pure

caching scheme, and setting it to 1 equivalent to basic flooding as messages are forwarded to all peers.

Figure 3 shows the effect of these configuration settings on the time taken by Pegasus to generate a plan. Comparing these times to those in Figure 2, we notice that execution time improves when the resource information is distributed over 8 nodes rather than a single index. This is because query processing is CPU intensive, and distributed information processing improves performance. We also see that query caching significantly reduces execution time by reducing the number of messages processed by the network. We note that network delays are not modeled in this experiment.

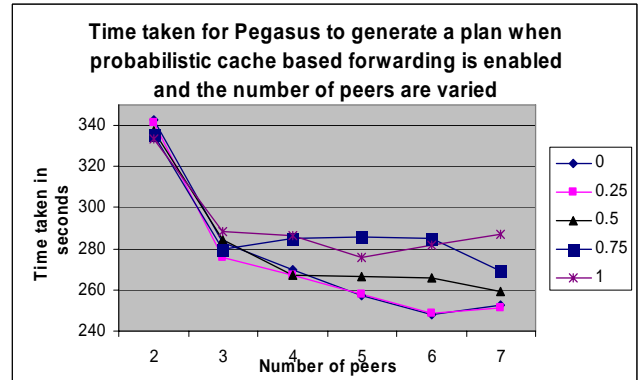


Figure 3 Effect of probabilistic forwarding on time taken by Pegasus to generate a plan

IV. CONCLUSIONS AND FUTURE WORK

We described the design and implementation of a P2P information system based on the GT4 Index Service. Our initial experiments with this system show that its performance is comparable to that of a centralized index and that optimizations like caching queries significantly reduce the number of messages sent in the system. We will continue testing our system with different query loads and applications in an effort to demonstrate the scalability and self-organization properties of the system.

ACKNOWLEDGMENT

We are grateful to Karan Vahi, Gaurang Mehta and Laura Pearlman for their help in integrating the GT4 Index Service and the P2P Index Service with Pegasus.

This research was supported in part by DOE Cooperative Agreements DE-FC02-01ER25449 & DE-FC02-01ER25453.

REFERENCES

- [1] Czajkowski, K., et al. (2004). *The WS-Resource Framework Version 1.0*. <http://www.globus.org/wsrfl/specs/ws-wsrf.pdf>
- [2] Deelman, E., et al. (2004). *Pegasus: Mapping Scientific Workflows onto the Grid*. Across Grids Conf., Nicosia, Cyprus.
- [3] Foster, I., et al. (2004). *Modeling Stateful Resources with Web Services version 1.0*. <http://www-128.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf>
- [4] Iammitchi, A., et al. (2002). *A Peer-to-Peer Approach to Resource Discovery in Grid Environments*. Eleventh IEEE Int'l. Symposium High Performance Distributed Computing (HPDC-11), Edinburgh, Scotland.