ON THE BENEFITS OF PROFILE-BASED NAMING FOR LARGE NETWORK COMPUTING SYSTEMS

Balasubramaneyam Maniymaran
Department of Electrical & Computer Eng.
McGill University
Montreal, QC H3A 2A7, Canada
Email: bmaniy@cs.mcgill.ca

Muthucumaru Maheswaran
School of Computer Science
McGill University
Montreal, QC H3A 2A7, Canada
Email: maheswar@cs.mcgill.ca

Abstract

Naming and discovery are two critical issues of a wide-area network computing system. The rising popularity of wide-area systems have resulted in the development of a variety of naming and discovery systems that are centered on a "descriptive" paradigm, where resources and services are described by a set of attribute-value tuples. This paper presents new evidence to suggest that descriptive naming systems of large-scale computing systems have special properties that can be exploited to construct very efficient and scalable implementations. We present the empirical evidence and perform simulations comparing a stock descriptive naming system with an enhanced naming system. Our simulation results indicate that the overhead caused by the naming system can be significantly reduced without any perceivable decline in performance.

1 Introduction

A naming system plays a crucial role in the success of the *network computing* (NC) system that is built on it. IP addresses, domain names, and uniform resource locators are example naming schemes that have successfully supported widely deployed NC systems. The development of various large-scale systems such Grid computing, utility computing, and pervasive computing are warranting another look at this well-studied topic. The major reason being their requirement for flexible, efficient, and scalable naming systems that can support resource and service discovery.

Existing naming systems can be classified into two major classes: *label*-based systems and *description*-based systems. Label-based systems affix a label, say a DNS name, with an object and use it to locate and access the object. The labels can be derived in different ways (for example, several peer-to-peer systems derive them from the contents of the object). Description-based systems, on the other hand, use a set of attribute-value tuples to name or to *describe* an object. Even though it provides flexibility in answering resource queries, it comes at the cost of additional overhead. Most part of the overhead is associated with maintaining databases of the attribute-value tuples, enforcing consistency among a network of such databases, resolving queries using values within these databases. Often the overhead increases dramatically with increasing size of the network [1].

Another significant difference between the two classes of systems is that the label-based systems normally result in unique names for the objects. The description-based system can result in non-unique names because the name is essentially a description of the object and the uniqueness is determined by the attributes used for description. The two classes of names are used for different purposes. The label-based names are used for locating and accessing a specific object. Whereas, description-based names are used for locating and accessing an object belonging to a given family of objects. Usually, this is an extension of a resource, data, or service discovery mechanism.

This paper proposes a new naming strategy called the *profile*-based naming that attempts to combine the benefits of description-based and label-based naming systems. In simple terms, profile-based naming considers the set of attribute-value pairs describing an object to be the profile of the object. The profiles are labeled with *type IDs* that become the name of the corresponding objects.

Although resources and queries are free to choose any combinations of attribute-value tuples, the profile-based naming is based on the argument that only a small subset of all possible combinations will be used disproportionately more than the rest. Such phenomena is observed in many different types of computer systems such as Internet topology [3], Weblogs [4], and webpages [5]. In Internet topology, this behavior is due to the fact that a newly joining node is likely to prefer adjacencies with already popular nodes. This rationale can be extended to naming systems as well. Clients of an NC system form their queries based on the requirements of the applications they want to execute and the feedback from previous users on the appropriateness of different resources. This makes certain attribute-value tuples to appear in a disproportionately large number of queries. The profile-based naming attempts to exploit this popularity skewness to select a minimal set of profiles while maximizing the fraction of queries that can be handled.

Section 2 presents a study on CNET.com database to support the argument behind the profile-based naming. Section 3 presents a conceptual view of naming systems

and introduces a profile-based naming system. Section 4 describes the simulation studies and the results obtained. Some related works are presented in Section 5 and the difference between existing work and our work is highlighted.

2 A Case for Reconsidering Naming

To proof the conjecture that the attribute-value tuple space for resource descriptions and queries in a description-based naming system is power-law distributed, we require significant amounts of trace data with appropriate levels of detail from existing naming systems. Unfortunately, existing trace data on current computing utilities, Grid computing systems, multi-cluster computing centers do not provide sufficient insights. For example, an examination of the resource characteristics of large-scale Grid deployments show that the resources have highly correlated attribute-value tuple descriptions [6, 7]. Most resources in these environments tend to have the same processor architecture to improve portability of code among the different resources to ease the resource management. Similarly, a review of the queries submitted to high-performance computing centers reveal that many requests leave a significant portion of the attributes as "don't care." This is partly because the computing centers have resources that are roughly equivalent [8]. Although these observations hint to a smaller attribute-value tuple subspace that is highly popular, the data is insufficient to draw any conclusions.

The lack of useful trace data from computing utilities prompted us to investigate other potential sources of information. One such source is `CNET.com` [2], which is an online computer system review site. We use this site to examine the resource diversity, which is one of the factors that determine the flexibility requirement of the naming systems. Our rationale for using data from `CNET.com` to examine resource diversity is as follows. The `CNET.com` review database has more reviews for systems that are expected to be popular among its clients. We assume that `CNET.com` would have gauged the relative popularity of the systems and populate their review database to maximize the information needs of their clientle. Extending this argument, we can even say that the popularity profile of a review database can foreshadow the popularity profile we can expect in a future population of computer systems.

The CNET review database uses many attributes to characterize a desktop of which four attributes are selected in this analysis. They are processor class, clock speed, memory size, and hard disk size. Others are deemed less important from the perspective of a computing utility or large-scale NC. The processor class attribute can have thirteen possible values and six values are possible for other attributes. This gives a total of 2808 possible types for the systems reviewed by CNET. The frequency of each type in those systems is counted and the resulting cumulative histogram is shown in Figure 1.

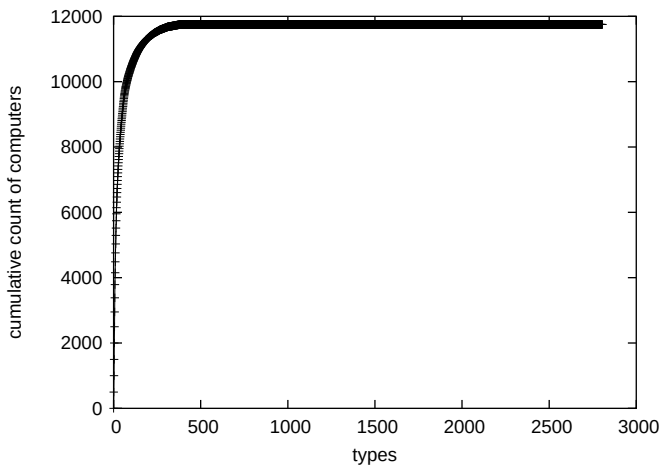

Figure 1. Cumulative histogram of desktops over number of types.

Several interesting numbers can be noted from the above figure. Only 110 types (3.6% of total types) are required to describe 90% of the total systems and only 383 types (13.6% of total types) are populated by the whole database (which has more than $10,000$ systems). These observations suggest that a smaller subspace of the attribute-value tuple space is popular. This could be exploited by a description-based naming system to improve its efficiency. One way of improving the efficiency by exploiting this property and the resulting benefits is the topic of the rest of this paper.

3 A New Naming Architecture

3.1 A Conceptual Look at Naming

An important component of a naming system is the *name space*. If we consider a description-based system with a bounded number of attributes and bounded distinct possible values for each attribute, the name space will be bounded. We refer to this name space as the *possible space* because all name specifiers should fall into this space. For example a description-based system with five attributes each having five possible values produces a possible space with $5^5 = 3125$ names. In general, with attributes that have continuous values, the possible space can be unbounded. The *populated space* refers to a sub space of possible space that consists of the names of resources that are actually present in the system. The *popular space* is a sub space made of the names that are referred to by the popular queries for some arbitrary popularity threshold. We introduce another sub space called the *perceived space* that is actively maintained by the discovery and dissemination system associated with the naming system. The nested organization of the different sub spaces is shown in Figure 2.

Although the popular space should be contained within the populated space, the perceived space is a design

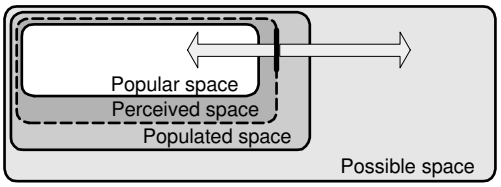

Figure 2. Nested organization of the different sub name spaces.

parameter and can be placed differently. The overhead in the system can be controlled with the expanse of the perceived space: (a) a naming system can immediately reject a query that lies outside the perceived space, which prevents the associated message and processing overhead and (b) when the perceived space is designed smaller than the populated space, message and processing overheads can be further reduced by preventing the dissemination of status information for unpopular resources.

3.2 A Profile-Based Naming System

Figure 3 illustrates the overall architecture of the *profile-based naming* system. Incoming resources are presented to the resource profiling module which tags them with type IDs appropriately. Once a resource is profiled, it is accepted by the profile-based naming system. Similarly, when queries arrive at the system, they are also profiled based on the same profile repository. Depending on the number of recognized profiles in the repository, a portion of incoming resources and queries will fail to be profiled into a type.

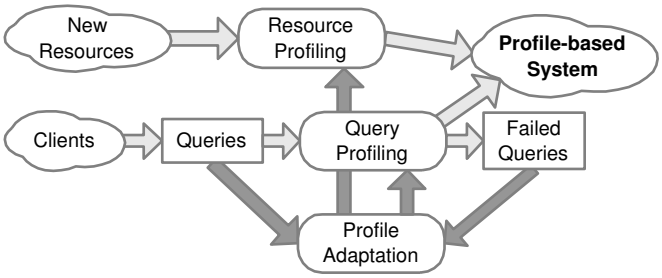

Figure 3. Architecture of a profile-based naming system.

The contents of the profile repository is tuned by the *profile adaptation* module such that the names within the most popular sub space are included in the repository. It can add new profiles and retire old profiles. Profile adaptation is basically a feedback system which monitors the incoming queries and their fate and increase the number of recognized types only to keep the number of failed queries below a design specified threshold. Profile adaptation does not take resource profiling failures into account because it is only the fate of queries that determines the performance of the system. However, as mentioned earlier, query patterns have an influence on the resource types that already have specific profiles. Therefore, if the query profiling failure is low, the resource profiling failure will be low as well. Removing an existing profile can be an involved process because all resources matching that profile need to be informed. One approach is to expire a profile after a given time and re-profile all resources with expired profiles.

Tagging resources and queries with type IDs reduces the system overhead in many ways: (a) compact resource dissemination messages; (b) compact query messages; (c) avoidance of information loss due to aggregation or summarization of resource descriptions; and (d) possibility of using peer-to-peer lookup services such as Pastry [9] in the resource discovery process as they can map labels efficiently.

With the restricted number of recognized profiles, there is always a possibility of failure of resource queries even though there are resources in the system that match the description. This is where the trade-off between performance and overhead comes in. However, the performance can be increased with a hybrid system where the profile-based system is implemented in a distributed manner with a centralized (small scale) description-based system to capture the resources and queries that are not profiled.

4 Simulation

4.1 Simulation Setup

The objective of the simulation study is to evaluate the performance differences between the description-based and profile-based naming systems. In the simulations, we assume that a resource can be described by five attributes and each attribute can have five values. The naming system is implemented by the name resolvers that are distributed across the wide-area network and interconnect using a peer-to-peer overlay. A resource connects to a particular name resolver to which it advertises itself and also submits naming queries. The *flooding* protocol is used for inter-resolver dissemination. To reduce the inter-resolver message sizes, the resolvers summarize the local information and disseminate only the digests. When a resolver determines that an incoming query cannot be satisfied by resources associated with the resolver, the resolver searches through summaries received from other resolvers to decide to which resolver(s) it should forward the query. The home resolver chooses the first reply it receives and ignores the rest. The queries are assumed to arrive in a Poisson distribution. One major difference between query and resource specifications is that in query specifications wild cards can be present. Attribute values are generated for the queries using a power-law like distribution. No correlation is assumed between the different attributes of a query.

Performance of the description-based and profile-based naming systems are simulated against the following parameters: (a) network size as (measured in number of

nodes), (b) number of resolvers, and (c) number of queries arriving at the system.

Description-based naming system:

This system identifies a resource by a set of attribute-value tuples. A resolver in this system receives a query that specifies either partially or completely the attribute-value pairs that should hold for the target resource. The resolver returns the address of one such target resource. The resolvers are uniformly distributed over the network such that a resource attaches to the closest resolver it could find. Because the resolvers disseminate the digest of the local resource characteristics, remote resolvers can forward some queries that cannot be satisfied locally due to false positives.

Profile-based naming system:

In this system, resource type IDs are used for dissemination as well as queries. Because a query can be expressed using wild-card attributes, a query can match multiple types. Profile-based system allows one resolver for each recognized type. However, when all the recognized types are populated by the resources in the system, the actual number of resolvers in the system will be low than the allowed number.

4.2 Simulation Results

4.2.1 Message overhead with number of resolvers

The inter-resolver dissemination and query forwarding/reply are main sources of message overhead in both systems. Figure 4 compares the message overhead for the two systems for $100,000$ queries and network sizes of $5,000$ and $10,000$ nodes for varying number of resolvers. The y-axis in the figure is in log scale. The results shows that the message overhead rises with the increasing number of resolvers. However, the increase is much low in the profile-based system. In Figure 5 shows the loading of the resolvers as messages per resolver. The loading is much less in the profile-based system.

Also the figures shows that the actual number of resolvers in the system in the profile-based system always stops growing beyond a certain limit (in this case 3%). It reflects the skewness in the popularity of the resource types. The interesting point here is that even with a fewer number of resolvers in the profile-based system, loading of the resolver is much less in this system than the other.

4.2.2 System performance with Network Size and Queries

In this section, we examine the variation of different performance measures with network sizes and queries. Figure 6 shows the variation of message overhead with different network sizes. For this experimentation, the number of queries is kept constant at $100,000$ and the number of resolvers is kept at 3% of the number of nodes in the network.

The low message overhead of profile-based system is due to two major reasons: (i) reduction of needless forwarding of queries and (ii) reduction in dissemination due to reduction in the number of resolvers. The query forwarding is more inefficient in description-base system because there can be a lot of false query forwards to unsuitable resolvers due to aggregation of resource description.

A message generated by a resolver in response to a query is considered *wasted* if it fails to find a resource that is not busy and satisfies the query constraints. Figure 7 shows the percentage of the forwarded messages that are wasted on either system. The large number of wasted forwarded messages in the description-based system is the result of both false positive forwarding and busy resources. On the contrary, in profile-based system it is only due to the busy resources.

Because a resolver generates several message in response to a query, wasted messages do not directly relate to the actual query success rate. In Figure 8, we measure the percentage of failed queries with network size. The figure shows that for larger network sizes the two schemes are equally successful and for smaller networks the description-based system outperforms the profile-based system by a small margin. In a profile-based system, a query can fail due to two reasons: (a) unable to resolve the query into one or more of the recognized types, and (b) all the resources of the corresponding type are busy.

We repeated the experiment with varying number of queries and fixed network size of $5,000$ nodes and fixed number of resolvers at 200. The results for overhead and wasted forwarded messages follow the similar trend as the results with varying network size. Figure 9 shows the result for percentage of failed queries. Even though the flexibility of description-based system satisfies more queries, the performance different is very small ($\approx$0.1%)

4.3 Scalability Analysis

A distributed system is considered scalable if it could scale up while maintaining high productivity per unit overhead [10]. Here we use *(message overhead)/(success rate)* as a measure of scalability. The message overhead is the total messages that are exchanged in the system. The success rate gives the rate at which the naming system is able to successfully resolve naming queries. In this experiment, the network size is increased and simultaneously the number of queries is increased. The number of queries is always ten times the number of nodes. The number of resolvers is kept at 5% of the number of nodes. The results shown in Figure 10 are normalized to the result of 500-node network and plotted against the normalized network size. From the figure it is evident that the scalability of the description-based system is much worse than that of the profile-based system. The scalability measure shown in Figure 10 is based on a flooding protocol. With a more efficient dis-

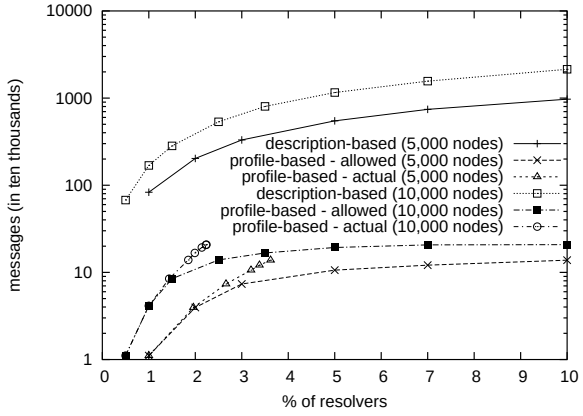

Figure 4. Message overhead with number of resolvers.

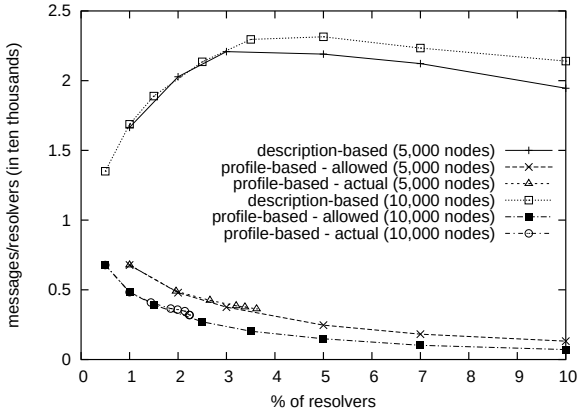

Figure 5. Resolver load with number of resolvers.

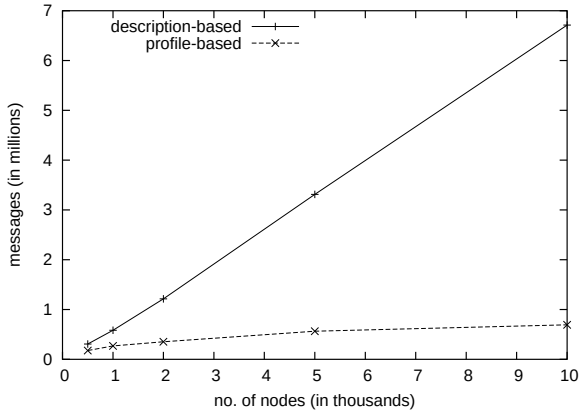

Figure 6. Message overhead with network size.

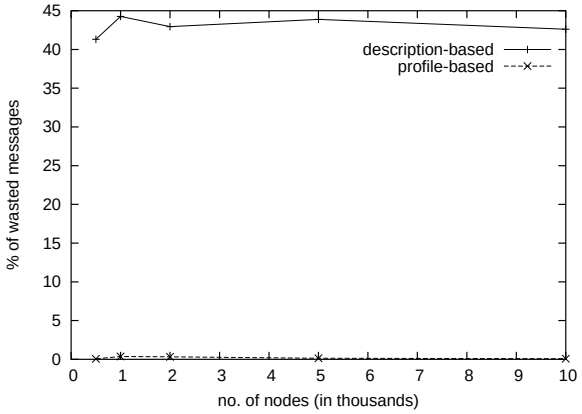

Figure 7. Percentage of wasted forwarded messages with network size.

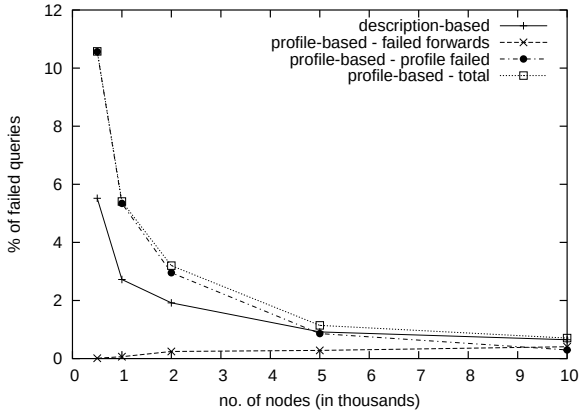

Figure 8. Percentage of unsuccessful queries with network size.

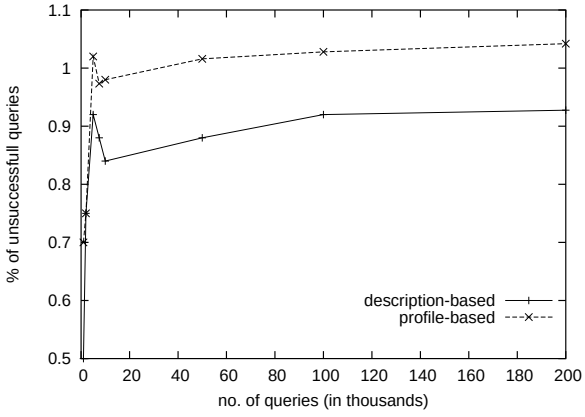

Figure 9. Percentage of unsuccessful queries with number of queries.

semination algorithm, the scalability of both protocols can be improved.

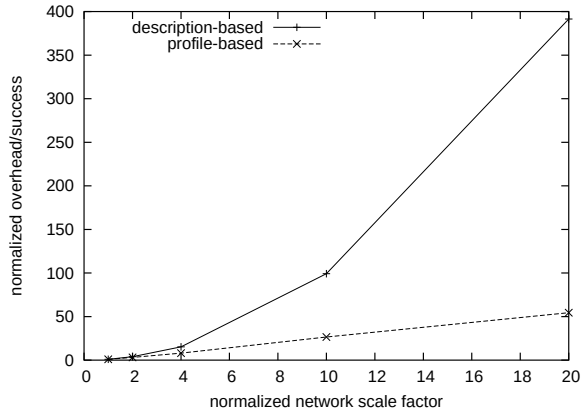

Figure 10. Variation of normalized overhead/success rate with normalized size.

5 Related Work

The *meta-directory services* (MDS-1) [11] was the backbone of Globus toolkit version 1.x [12]. MDS-1 extends LDAP so that multiple information sources can update a particular entry in the system. In Globus toolkit version 2, the *monitoring and discovery services* (MDS-2) [13] takes over the functionality of MDS-1 to address some of the performance bottlenecks associated with MDS-1 that were inherited from LDAP. In MDS-2, flexibility in discovery is achieved using *aggregate directory services* which can produce customized resource views/collections from the information received from low-level *information providers* that are analogous to the LDAP servers. The aggregate directory servers are *virtual organization* (VO) specific, a design aspect that increases scalability. The idea of having aggregate services supports our concept of profiling: aggregate services are designed to handle popular query types. In MDS-2, the indirection between aggregate services and information providers, reduces the responsiveness of the system, which is addressed by introducing resource information caching at the aggregate servers [14]. When up-to-date information is required, it should be pulled from the information providers on a query-by-query basis [13]. Our approach differs fundamentally from MDS. Where MDS uses aggregation to make a pure description-based system scalable and flexible, we take a different approach of converting a description-based system to label-based system. As network sizes scale up to include millions of resources (merger of peer-to-peer and Grids), our approach has significant advantages.

The *relational grid information services* (RGIS) [15, 1] is an implementation of a Grid information service using relational database technology, where resource advertisements are considered as record insertions and resource requests as database queries. The focus of this research is to develop efficient strategies to use relational database technologies towards achieving a high performance naming system.

The *intentional naming system* (INS) [16] is a naming system that is based on the "intent" of the applications rather than the network locations. The names in INS are formed as a list of attribute-value tuples, where the sequence of attribute appearance determines the search process for the resources. INS allows wild-cards values in queries. *Twine* [17] is the dissemination/discovery architecture based on INS for pervasive computing applications.

The *lightweight directory access protocol* (LDAP) [18] is one of the early works in naming for resource discovery systems. It is a client–server based directory service where the name entries are distributed among hierarchically organized name servers. The name entries are made up of a set of attribute-value tuples that can be used to name entities ranging from people to computers. The read-optimized nature of the LDAP directories restricts its applicability in a highly dynamic wide-area system.

6 Conclusions and Future Work

This paper proposes a new naming strategy called the profile-based naming that attempts to combine the benefits of description-based and label-based naming systems. In simple terms, profile-based naming considers the set of attribute-value pairs describing an object to be the profile of the object. The profiles are labeled with type IDs that become the name of the corresponding objects. We presented empirical data that suggests a description-based naming system might have a smaller subset of names that form a popular sub space of the whole possible naming space. Significant amount of data from operational description-based naming systems is necessary to conclusively proof or disprove this theory.

Assuming that some form of power-law relationship holds for the description-based naming systems, we performed large-scale simulation studies to compare profile-based and description-based naming systems. The simulation results indicate that profile-based naming can significantly reduce the message and processing overheads without adversely impacting the success rate of the name resolution process. One of the major reasons for the overhead reduction in the profile-based naming is the elimination of erroneous query message forwarding. This happens in description-based systems because the query messages are forwarded based on the resource aggregates disseminated by the different resolvers in the description-based system. Other sources of message overhead reduction include reductions in status dissemination messages due to the exclusion of unpopular resources.

In summary, the work presented in this paper is novel and significant for the following reasons:

- It provides the first known evidence that a descriptive

name space can potentially have a popular sub space that is used by a disproportionately large number of name queries. Although it might seem that caching already exploits this property, it does not fully exploit it. For instance, caching remembers a particular binding that is obtained from the resolution process for a given name specifier. In contrast, profile-based naming identifies the popular name specifiers and builds a lean and efficient naming system that obviates the necessity for caching. This also makes the profile-based naming "friendly" towards dynamic parameters because the name query is resolved at real-time without depending on cached values.

- It presents a profiling process that turns a description-based name specifier into a label. With the advent of highly scalable self-organizing peer-to-peer lookup services such as Pastry and Chord, a label based name specifier can be handled efficiently in a scalable manner. Further, P2P lookup services such as Pastry already handle network proximity, which could be exploited by the profile-based naming system.
- It describes an architecture for deploying profile-based naming system in conjunction with an existing description-based naming system.
- It shows the significant performance differences that exist between description-based and profile-based naming systems.

We are continuing this study by examining various aspects of profile-based naming. Some of them include: (a) introducing multiple profiling agents and (b) context specific profiling.

References

[1] P. Dinda and B. Plale, "A unified relational approach to grid information services," Grid Forum Informational Draft GWD-GIS-012-1, Feb. 2001.

[2] "CNET.com web page, Desktop reviews," http://www.cnet.com.

[3] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the Internet topology," in *ACM SIGCOMM*, 1999, pp. 251–262.

[4] C. Shirky, "Power laws, weblogs, and inequality," http://shirky.com/ writings/powerlaw_weblog.html, Feb. 2003, On-line article at Clay Shirky's Writings About the Internet.

[5] A.L. Barabasi, R. Albert, and H. Jeong, "Scale-free characteristics of random networks: The topology of the World Wide Web," *Physica A 281*, pp. 69–77, 2000.

[6] "Information Power Grid," http://www.ipg.nasa.gov/.

[7] "University of Victoria Grid test bed," http://grid.phys.uvic.ca/.

[8] "SuperCluster.org," http://www.supercluster.org/.

[9] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Nov. 2001, pp. 329–350.

[10] P. Jogalekar and M. Woodside, "Evaluating the scalability of distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 6, pp. 589–603, Jan. 2000.

[11] S. Fitzgerald, I. Foster, C. Kesselman, G. Von Laszewski, W. Smith, and S. Tuecke, "A directory service for configuring high-performance distributed computations," in *Proceedings of 6^{th} IEEE Symp. on High Performance Distributed Computing*. 1997, pp. 365–375, IEEE Computer Society Press.

[12] "The Globus alliance," http://www.globus.org/.

[13] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman, "Grid information services for distributed resource sharing," in *The 10^{th} IEEE International Symposium On High-Performance Distributed Computing*, Aug. 2001.

[14] X. Zhang, J. Freschl, and J. Schopf, "A performance study of monitoring and information services for distributed systems," in *The 12^th International Symposium on High-Performance Distributed Computing*, Aug. 2003, pp. 270–282.

[15] P. A Dinda and D. Lu, "Nondeterministic queries in a relational grid information service," Technical report, Computer Science Department, Northwestern University, Apr. 2003.

[16] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley, "The design and implementation of an intentional naming system," in *The 17^{th} ACM Symposium on Operating Systems Principles*, Dec. 1999, pp. 186–201.

[17] M. Balazinska, H. Balakrishnan, and D. Karger, "INS/Twine: A scalable peer-to-peer architecture for intentional resource discovery," in *International Conference on Pervasive Computing*, Zurich, Switzerland, Aug. 2002.

[18] T. A. Howes, "The lightweight directory access protocol: X.500 lite," Technical report, Center for Information Technology Integration, University of Michigan, July 1995.