INFORMATION GATHERING AND DISTRIBUTION IN NOMENCLATOR

Joann J. Ordille
joann@research.att.com

AT&T Bell Laboratories
Computing Science Research Center
600 Mountain Ave., Rm. 2C-301
Murray Hill, NJ. 07974

1. INTRODUCTION

As the Internet grows beyond a million hosts in tens of thousands of organizations, it is increasingly difficult to locate any particular object. Graph-based information services, like the World-Wide Web [1] and X.500 [2], are frustrating, because users must guess the unique names for objects or navigate the information space to find a particular object. Descriptive (i.e. relational) information services offer the promise of simple information access through a non-procedural query language. Users locate information by describing attributes of the information.

The key to speed in descriptive query processing is constraining the search space using two new techniques, called an *active catalog* and *meta-data caching*. The active catalog constrains the search space for a query by returning a list of data repositories where the answer to the query is likely to be found. Components of the catalog are distributed indices that isolate queries to parts of the network, and smart algorithms for limiting the search space by using semantic, syntactic, or structural constraints. Meta-data caching improves performance by keeping frequently used characterizations of the search space close to the user, thus reducing active catalog communication and processing costs. When searching for query responses, these techniques improve query performance by contacting only the data repositories likely to have actual responses, resulting in acceptable search times.

An active catalog and meta-data caching are used in a prototype descriptive name service, called *Nomenclator*. Nomenclator answers queries about people by retrieving information from a variety of name services, including CCSO name services [3], and relational database services. A previous version of Nomenclator improved the performance of descriptive queries in X.500 [7].

The Nomenclator effort is a continuing research program to develop techniques that improve query speed, preserve organization autonomy, protect privacy and enhance data availability in large-scale information systems [7], [6], [5], [4]. This paper explores aspects of Nomenclator that relate to distributed, heterogeneous information gathering. Section 2 identifies two information gathering principles that are central to the active catalog. Section 3 briefly describes techniques for meta-data gathering, organization and distribution in Nomenclator. Section 4 summarizes previous performance results for Nomenclator. Finally, Section 5 describes areas where Nomenclator could be enhanced by more intelligence, or could enhance existing intelligent systems.

2. INFORMATION GATHERING PRINCIPLES

The active catalog is a mechanism for isolating queries within a subset of the data repositories that store a relation. The active catalog constrains the search space for a query, eliminating the overhead of contacting data repositories that will not contribute to the query answer. Nomenclator implements the active catalog with a distributed catalog service and a query resolver. The distributed catalog service gathers meta-data about data repositories and makes it available to data users. Meta-data includes constraints on attribute values at a data repository, known patterns of data distribution across several data repositories, search and navigation techniques, schema and protocol translation techniques, and the differing schema at data repositories. Query resolvers at the user sites retrieve, use, cache, and re-use this meta-data in answering user queries. The catalog is *active* in two ways. First, some meta-data moves from the distributed catalog service to each query resolver during query processing. Second, the query resolver uses the initial meta-data, in particular the search and navigation techniques, to generate additional meta-data that guides query processing. Typically, one resolver process serves a few hundred users in an organization, so users can benefit from larger resolver caches.

Two information gathering principles are central to the active catalog. The first principle relieves the users of the need to understand the heterogeneous structures of the information space. The second principle promotes scale in distributed query processing by decentralizing and replicating Nomenclator's knowledge about the structure of the information space.

Principle 1: Owners of data provide informa-

tion that guides query processing.

Owners make their data known to users by supplying Nomenclator with information about the location, format, contents, and protocols of their data repositories. Nomenclator differs from the many systems that expect the user to provide information about the location of data, or to navigate the information space to locate information. Experience with Nomenclator shows that gathering a small amount of information from data owners can have a substantial positive impact on the ability of users to retrieve information. For example, each owner of a CCSO data repository provides a mapping from the schema at the owner's data repository to Nomenclator's global `People` schema and the possible values for any attributes with small domains at the data repository (such as the `city` or `state` attributes). With this information, Nomenclator can isolate queries to a small percentage of the 270 CCSO data repositories, and provide an integrated view of their data. Nomenclator provides tools that minimize the effort that data owners expend in characterizing their data repositories.

> *Principle 2:* Query resolvers gather information, as a by-product of query processing, that can be re-used to improve performance.

Query resolvers cache techniques for constraining the search space and the results of previously constrained searches (meta-data), and past query answers (data) to speed future query processing. Meta-data and data caching tailor the query resolver to the specific needs of the users at the query site. They also increase the scale of a Nomenclator system by reducing the load from repeated searches or queries on the distributed catalog service, data repositories, and communications network.

3. META-DATA TECHNIQUES

The active catalog structures the information space into a collection of relations about people, hosts, organizations, services and other objects. It collects meta-data for each relation and structures it into *access functions* for locating and retrieving data. Access functions respond to the question: "Where is data to answer this query?" There are two types of responses corresponding to the two types of access functions. The first type of response is: "Look over there." *Catalog functions* return this response; they constrain the query search by limiting the data repositories contacted to those having data relevant to the query. Catalog functions return a referral to data access functions that will answer the query or to additional catalog functions to contact for more detailed information. The second response to "Where?" is: "Here it is!" *Data access functions* return this response; they understand how to obtain query answers from specific data repositories. They return tuples that answer the query. Nomenclator supplies access functions for common name services, and organizations can write and supply access functions for data on their repositories.

Access functions are implemented as remote or local services. Remote access functions are services that are available through a standard remote procedure call interface. Local access functions are functions that are supplied with the query resolver. Local access functions can be applied to a variety of indexing and data retrieval tasks by loading them with meta-data stored in distributed catalog service. Remote access functions are preferred over local ones when the resources of the query resolver are inadequate to support the access function. The owners of data may also choose to supply remote access functions for privacy reasons if their access functions use proprietary information or algorithms. Local functions are preferred whenever possible, because they are highly replicated in resolver caches. They can reduce system and network load by bringing the resources of the active catalog directly to the users.

Remote access functions are simple to add to Nomenclator and local access functions are simple to apply to new data repositories, because the active catalog provides *referrals* that describe the conditions for using access functions. Each referral contains a template and a list of references to access functions. The template is a conjunctive selection predicate that describes the scope of the access functions. Conjunctive queries that are within the scope of the template can be answered with the referral. When a template contains a wildcard value ("*") for an attribute, the attribute must be present in any queries that are processed by the referral. The system follows the following rule:

> *Query Coverage Rule:* if the set of tuples satisfying the selection predicate in a query is covered by ($\subseteq$) the set of tuples satisfying the template, then the query can be answered by the access functions in the reference list of the referral.

For example, the query below:

```
(c = "US" and name = "Ordille")
```

is covered by the following templates in Lines (1) through (3), but not by the templates in Lines (4) and (5):

```
(1) (c = "US" and name = "*")

(2) (c = "US" and name = "Ordille")
```

(3) (c = "US")

(4) (o = "*")

(5) (c = "US" and name = "Inners")

Referrals form a generalization/specialization graph for a relation called a *referral graph*. Referral graphs are a conceptual tool that guides the integration of different catalog functions into our system and that supplies a basis for catalog function construction and query processing. A *referral graph* is a partial ordering of the referrals for a relation. It is constructed using the subset/superset relationship: $S \subset G$. A referral S is a subset of referral G if the set of queries covered by the template of S is a subset of the set of queries covered by the template of G. S is considered a *more specific* referral than G; G is considered a *more general* referral than S. For example, the subset relationship exists between the pairs of referrals with the templates listed below:

```
(1) (c = "US" and name = "Ordille")
    ⊂ (c = "US")

(2) (c = "US" and name = "Ordille")
    ⊂ (c = "US" and name = "*")

(3) (c = "US" and name = "*")
    ⊂ (c="US")

(4) (c = "US") ⊂ ()
```

but it does not exist between the pairs of referrals with the following templates:

```
(5) (c = "US"), (dept = "CS")

(6) (c = "US" and name = "Ordille"),
    (c = "US" and name = "Inners")
```

In Lines (1) and (2), the more general referral covers more queries, because it covers queries that list different values for name. In Line (3), the more general referral covers more queries, because it covers queries that do not constrain name to a value. In Line (4), the specific referral covers only those queries that constrain the country to "US" while the empty template covers all queries.

During query processing, wildcards in a template are replaced with the value of the corresponding attribute in the query. For any query covered by two referrals S and G such that $S \subset G$, the set of tuples satisfying the template in S is covered by ($\subseteq$) the set of tuples satisfying the template in G. S is used to process the query, because it provides the more constrained (and faster) search space. The referral S has a more constrained logical search space than G, because the set of tuples in the scope of S is no larger, and often smaller, than the set in the scope of G. Moreover, S has a more constrained physical search space than G, because the data repositories that must contacted for answers to S must also be contacted for answers to G, but additional data repositories may need to be contacted to answer G.

In constraining a query, a catalog function always produces a referral that is more specific than the referral containing the catalog function. Wildcards ("*") in a template indicate which attribute values are used by the associated catalog function to generate a more specific referral. In other words, catalog functions always follow the rule:

> *Catalog Function Constrained Search Rule*: Given a referral R with a template t and a catalog function cf, and a query $q \subseteq t$, the result of using cf to process q, cf(q), is a referral R′ with template t′ such that $q \subseteq t'$ and $R' \subset R$.

Catalog functions make it possible to import a portion of the indices for the information space into the query resolver. Since they generate referrals, the resolver can cache the most useful referrals for a relation and call the catalog function as needed to generate new referrals.

The resolver query processing algorithm obtains an initial set of referrals from the distributed catalog service. It then navigates the referral graph, calling catalog functions as necessary to obtain additional referrals that narrow the search space. Sometimes, two referrals that cover the query have the relationship of general to specific to each other. The resolver eliminates unnecessary access function processing by using only the most specific referral along each path of the referral graph. The search space for the query is initially set to all the data repositories in the relation. As the resolver receives referrals to only data access functions, it forms their intersection to constrain the search space. The intersection of the referrals includes only those data repositories listed in both referrals. Intersection combines independent paths through the referral graph to derive benefit from indices on different attributes.

4. SCALE AND PERFORMANCE

Three performance studies of active catalog and meta-data caching techniques are available [4]. The first study shows that the active catalog and meta-data caching can constrain the search effectively in a real environment,

the X.500 name space. The second study examined the performance of an active catalog and meta-data caching for single users on a local area network. The experiments showed that the techniques to eliminate data repositories from the search space can dramatically improve response time. Response times improve, because latency is reduced. The reduction of latency in communications and processing is critical to large-scale descriptive query optimization. The experiments also showed that an active catalog is the most significant contributor to better response time in a system with low load, and that meta-data caching functions to reduce the load on the system. The third study used an analytical model to evaluate the performance and scaling of these techniques for a large Internet environment. It showed that meta-data caching plays an essential role in scaling the distributed catalog service to millions of users. It also showed that constraining the search space with an active catalog contributes significantly to scaling data repositories to millions of users. Replication and data caching also contribute to the scale of the system in a large Internet environment.

5. ARTIFICIAL INTELLIGENCE POSSIBILITIES

Nomenclator provides an additional layer of information service over the centralized, hierarchical or graph-based services of today. It can serve as a platform for more intelligent systems by locating an initial search space or collection of responses. Intelligent search agents can use the Nomenclator search spaces as a road map to locations where browsing is likely to be fruitful. Other intelligent agents can apply more sophisticated filtering and summarization techniques to the responses from Nomenclator queries.

The Nomenclator system architecture provides an approach to structuring any system that gathers and distributes meta-data to improve query performance. One possibility for enhancing the architecture is to augment the language used to specify referral templates and the partial ordering in referral graphs with conceptual hierarchies. The referral graph would then range from more general to more specific concepts, as well as from more general to more specific logical predicates. Information in data repositories that is now mapped to one vocabulary could be maintained in all its gradations of meaning. Users could obtain information about more precise concepts, and narrow or widen their searches by refining or generalizing the concepts in their queries.

6. REFERENCES

1. Berners-Lee, T., Caillau, R., Groff, J., and Pollermann, B. World-Wide Web: The Information Universe. *Electronic Networking: Research, Applications and Policy 1*, 2 (Spring 1992).
2. CCITT. The Directory. Recommendations X.500, X.501, X.509, X.511, X.518-X.521, 1988.
3. Dorner, S. and Pomes, P. The CCSO Nameserver – A Description. Technical Report, Computer and Communications Services Office, University of Illinois, August 1992.
4. Ordille, J.J. Descriptive Name Services for Large Internets. Ph.D. Thesis, University of Wisconsin, November 1993.
5. Ordille, J.J. and Miller, B.P. Database Challenges in Global Information Systems. *ACM SIGMOD International Conference on Management of Data* (May 1993), 403-407.
6. Ordille, J.J. and Miller, B.P. Distributed Active Catalogs and Meta-Data Caching in Descriptive Name Services. *Thirteenth International IEEE Conference on Distributed Computing Systems* (May 1993), 120-129.
7. Ordille, J.J. and Miller, B.P. Nomenclator Descriptive Query Optimization in Large X.500 Environments. *ACM SIGCOMM Symposium on Communications Architectures and Protocols* (September, 1991), 185-196.