BODHI: A Database Habitat for Bio-diversity Information

Srikanta J. Bedathur Abhijit Kadlag Jayant R. Haritsa*
Database Systems Lab, SERC/CSA
Indian Institute of Science, Bangalore 560012, INDIA
{srikanta, abhijit, haritsa}@dsl.serc.iisc.ernet.in

1. INTRODUCTION

Modern biodiversity research involves systematic and simultaneous study of macro- and micro-level relationships between various biological entities. Multi-domain queries of the following kind are increasingly common among the researchers in this field:

> *Retrieve the names of all plant species that have common infloresence characteristics, share a part of their habitats, and have a high chromosomal DNA sequence similarity with Michelia-champa*[1].

Answering this query requires the ability to process data across: (a) taxonomy hierarchies (*common infloresence*), (b) recorded spatial distribution of species (*common habitat*), and (c) genomic sequences (*chromosomal DNA sequence similarity*). Unfortunately, due to the lack of holistic database systems, biologists are often forced to split the query into component queries, each of which can be processed separately over specialized independent tools and services. Further, the individual results have to be combined either manually or through the use of a customized tool.

Motivated by this lacuna of an information management system that can support complex queries common to biodiversity research, we have recently built **BODHI** (Biodiversity Object Database arcHItecture), a native object-oriented database system that seamlessly integrates multiple types of data occurring in biodiversity studies. The BODHI system expresses the sample multi-domain query presented above using an OQL syntax as shown in Figure 1. To the best of our knowledge, BODHI is the first system to provide such an integrated view of diverse biological domains ranging from molecular to organism-level information.

In addition to providing a functionally comprehensive query interface, BODHI achieves high performance by employing a variety of specialized access structures reported in the research literature for handling predicates over taxonomy hierarchies and spatial data. While these index structures are efficient in their respective domains, a performance evaluation of BODHI indicated that the the costliest operators to handle are the sequence similarity queries which severely affect the performance of cross-domain queries [2]. Therefore, BODHI integrates SPINE, a space efficient sequence index structure [3], and TOP-Q [1], a high performance buffer management policy, for persistent suffix-tree construction and querying, leading to sub-minute evaluation of cross-domain queries involving sequence predicates, even on off-the-shelf Pentium-III Linux workstations.

```
SELECT species2.name FROM
   species1 IN PlantSpecies, species2 IN PlantSpecies,
   dna1 IN species1.DNAEntries, dna2 IN species2.DNAEntries
WHERE
species1.name = "Michelia-champa" AND
species1.flowerchar.inflochar = species2.flowerchar.inflochar AND
species1.georegion OVERLAPS species2.georegion AND
dna1 BLAST dna2 WITHIN 70;
```

Figure 1: Expressing a Multi-domain Query in BODHI

The system has a web-based graphical user interface for querying the database as well as for visualization of results. Easy dissemination of information is supported through the use of XML in publishing the results of queries, providing added semantics for future data exchange requirements.

2. ARCHITECTURE OF BODHI

The overall architecture of BODHI is shown in Figure 2. The SHORE storage manager at the base provides the fundamental needs of a database server such as device and storage management, transaction processing, logging and recovery management. The application specific modules, which supply the object, spatial and genomic services, are built over this storage manager and form the functional core of the system. The λ-DB extensible rule-based query processor and optimizer interfaces with these functional modules and performs query processing and produces efficient execution plans using the metadata exported by the modules. BODHI supports full OQL/ODL query and data modeling interface for creation of new database schemas, data manipulation and querying. Finally, the client interface framework and XML publishing engine form the external interface to BODHI.

The BODHI server is partitioned into three service modules: *Object*, *Spatial*, and *Sequence*, each handling the associated data domain. The service modules provide appropriate storage, a modeling interface, and evaluation algorithms for predicates over the corresponding data types.

Object Services. In querying over bio-diversity data, it is common to specify predicates over long relationship paths, or over an inheritance hierarchy rooted at a chosen base type. To efficiently handle these predicates, access methods for both inheritance

*Supported in part by a Swarnajayanti Fellowship from the Dept. of Science & Technology, Govt. of India, and a research grant from the Dept. of Bio-technology, Govt. of India.

[1]A fragrant medicinal plant endemic to India and Nepal.

SIGMOD 2004 June 13-18, 2004, Paris, France.

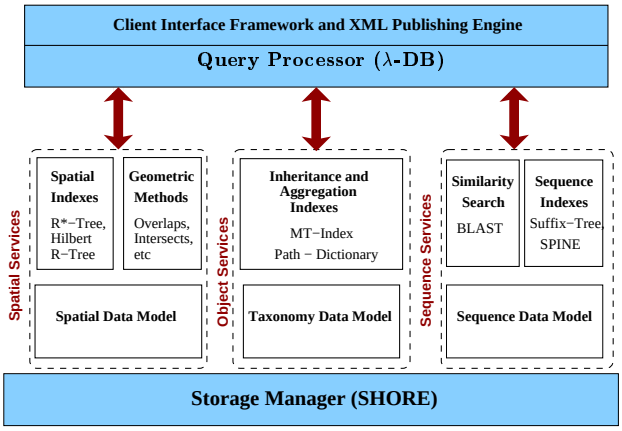

Figure 2: The Architecture of BODHI

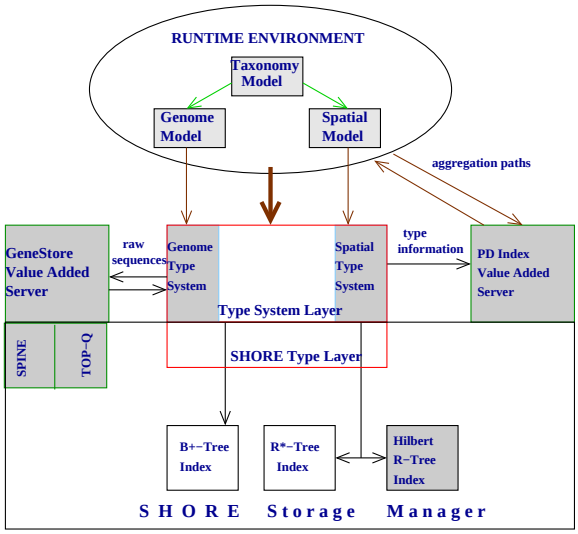

Figure 3: Implementation Schematic of BODHI

(*multi-key type Index*) and aggregation hierarchies (*path-dictionary index*) are included in this module.

Spatial Services. This module provides a spatial type system for modeling of spatial data associated with biological information. Various geometric operators such as *overlap*, *adjacent*, *area*, etc., are implemented over this type system. The module incorporates *R*-Tree* and *Hilbert R-Tree* indexing to speed up these otherwise expensive operators.

Sequence Services. This module provides efficient storage and operations over genome sequence data of species. It implements the de-facto standard *alignment-based* sequence similarity algorithms of BLAST and Smith-Waterman dynamic programming. To alleviate the response time bottleneck due to brute force scan adopted by these algorithms, this module incorporates a TOP-Q-buffered persistent suffix tree and a SPINE index, both of which have been developed in our recent research.

3. IMPLEMENTATION OF BODHI

The schematic in Figure 3 shows the placement of various components of BODHI in the overall system implementation. The gray filled boxes in the figure indicate the significant enhancements and features added to the public-domain components used in BODHI. These include:

Extended OQL for Novel Operators. With the support for spatial and sequence data domains, the OQL syntax is augmented with addition of new operators such as `OVERLAPS`, `ADJACENT`, `BLAST`, etc. Accordingly, the query optimizer is also significantly enhanced to generate efficient plans for predicates involving these special operators.

Support for Persistent Trie Indices. As part of the sequence services module, BODHI provides both vertically compacted (suffix-tree) and horizontally compacted (Spine) trie index structures for biological sequences. Further, the construction and query performance of these index structures is improved by providing a specialized buffering strategy called TOP-Q, as part of the buffer management module within SHORE.

Storage Manager Extensions. The SHORE storage manager provides a Value Added Server (VAS) framework for extending the functionality of the database server. The *PD-Index VAS* implements the path-dictionary index for the object services module, and the *GeneStore VAS* provides storage and retrieval of DNA and Protein sequences for the sequence services module.

Compact Storage of Sequence Data. Instead of storing the DNA sequences as character strings, BODHI stores them in a compressed form and performs queries over the compressed records rather than on the character strings.

Enhanced Path-dictionary Index. A typical biodiversity data model involves a number of aggregations with N:M cardinalities. The path-dictionary implementation in BODHI extends the original specification, which was restricted to 1:N cardinalities, to handle such relationship paths.

Efficient Hilbert R-Tree Implementation. To have page-level storage control, Hilbert R-Tree is implemented as a first class index structure within SHORE storage manager, by refactoring the existing R*-Tree implementation. To the best of our knowledge, this is the first implementation of Hilbert R-Tree index at this layer.

4. DEMONSTRATION

This demonstration aims at showing the capability of BODHI to efficiently process multi-domain biodiversity queries. It involves the formulation of complex queries using the browser-based interface and the rendering of XML output. The XML output will be shown using HTML, generated by applying appropriate XSL formatters.

5. REFERENCES

[1] S. Bedathur and J. Haritsa. Engineering a Fast Online Persistent Suffix Tree Construction. In *Proc. of the 20th IEEE Intl. Conf. on Data Engineering (ICDE)*, 2004.

[2] S. Bedathur, J. Haritsa, and U. Sen. The Building of BODHI, a Bio-diversity Database System. *Information Systems*, 28(4), 2003.

[3] N. Neelapala, R. Mittal, and J. Haritsa. SPINE: Putting Backbone into String Indexing. In *Proc. of the 20th IEEE Intl. Conf. on Data Engineering (ICDE)*, 2004.