CSE 718: Advanced Topics in Database Systems

Indexing Dataspaces

Xin Dong Alon Haley

Presented by: Sravanth Palepu

spalepu@buffalo.edu

Overview:

This paper proposes a novel way of indexing 'Dataspaces'. Dataspaces are collections of heterogeneous and partially unstructured data. It emphasizes on queries specifying varying degrees of structure, spanning keyword queries to more structure-aware queries.

Existing methods either build a separate index for each attribute in each data source to support structured queries on structured data, or create an inverted list to support keyword search on unstructured data. Consequently, they fall short in the context of queries that combine keywords and structure.

This paper discusses an idea by which indexing captures both text values and structural information using an extended inverted list. The index augments the text terms in the inverted list with labels denoting the structural aspects of the data like attribute tags and associations between data items.

Data from different data sources is modeled universally as a set of triples, which are referred to as a 'triple base'. Each triple is either of the form (instance, attribute, value) or (instance, association, instance).

In practice, users can specify predicate queries in two ways:

1. They can specify a query through a user interface featuring drop-down menus that show all existing attribute or association labels.

2. They can compose the query in certain syntax (such as the one shown in above example), specifying attribute or association labels that they know (such as those in data sources familiar to them).

This model captures attribute names with the indexed keywords to save both index space and lookup time. Whenever the keyword k appears in a value of the 'a' attribute, there is a row in the inverted list for k//a//. The cell (k//a//, I) records the number of occurrences of k in I's 'a' attributes. Similarly, suppose the instance I has an association r with instances $l_1, \ldots, l_n$ in the triple base, and each of $l_1, \ldots, l_n$ has the keyword k in one of its attribute values, the inverted list will have a row for k//r// and a column I

They introduced an inverted list that captures both attribute and association information. It is called an attribute-association inverted list (AAIL).

For indexing hierarchies, the paper came up with two solutions which is later combined them into one. First solution duplicates a row that includes an attribute name for each of its ancestors in the hierarchy. Second solution does not affect the number of rows in the inverted list. Instead, the keyword in every row includes the entire hierarchy path. These two solutions were combined to form a hybrid hierarchical index (KIL).

The efficiency of the hybrid hierarchical index (KIL) was tested. It takes 11.6 minutes to build the KIL and its size is 15.2MB. On average it took 15.2 milliseconds to answer a predicate query with no more than 5 clauses, and took 224.3 milliseconds to answer a neighborhood keyword query with no more than 5 keywords.

<u>Detailed Comments:</u>

- This model is much more that a simple Information Retrieval system
- The paper does not give a clear view of how a schema is developed of the Dataspaces
- It has a good mix of IR techniques and relational database techniques
- Not clear in explaining how the associations are defined or how attributes are declared
- The idea was expressed clearly and in an unambiguous manner

The two bodies of work most close to the proposed approach are indexing XML and on keyword queries in relational databases. There have been many indexing algorithms proposed for answering XML queries. They can be categorized into three classes: indexing on structure, indexing on value, and indexing on both.

The approach of the paper is different from the ones mentioned above in that it does not rely on any specific data model, and it uses one single index to capture both structure information and text values. In this way, this method is more oriented to keyword search, can more easily explore associations between data items, and can more efficiently answer keyword queries with simple structure specifications.

These were the various questions posed during the seminar:

1. Does the dataspace consist of different data models? If so, how do they form a single schema of those?
2. Do they use any Natural Language Processing for the query?
3. How do we find attributes and associations?
4. With reference to Figure 2 in the paper, will ‘yahoo’ return instance C1?
5. With reference to table 7 in the paper, is row 1 shadowing other rows?
6. What is the significance of threshold in Hybrid-ATIL?
7. What kind of data was used in the experiments and what is its size?