XML Data Partitioning for Parallel Holistic Twig Join Processing

Imam MACHDI†, Toshiyuki AMAGASA‡and Hiroyuki KITAGAWA‡

†Department of Computer Science, Graduate School of Systems and Information Engineering
‡Center for Computational Sciences, University of Tsukuba
†machdi@kde.cs.tsukuba.ac.jp, ‡{amagasa,kitagawa}@cs.tsukuba.ac.jp

1 Introduction

As the amount of XML data increases, parallel techniques become an attractive research area to give reasonable performance of XML query processing. However, one of the problematic issues in the design of parallel shared-nothing cluster systems is data partitioning strategy. Our system contains many XML documents with varying sizes of ranging from several KBs to several hundreds of MBs, but in total it could achieve TBs of XML data. Also, the system comprises a set of historical queries along with their frequency of occurrences recorded during past execution. Distributing those XML documents into cluster machines to achieve good workload balance and good performance of query processing is, in fact, a combinatorial optimization problem.

In this research, we study an XML data partitioning technique based on a multidimensional data model for parallel twig join processing. Some statistics extracted from XML data and query twig patterns are organized in this model and some operations on certain dimensions of the model are performed to define and refine partitions. Each partition is associated with a cost that is computed by our defined cost function. Based on costs of partitions we distribute related XML data to our cluster machines aiming at the entire cost balance and good performance of parallel query execution. Our cluster machines contains one coordinator for distributing queries and gathering query results and other machines performing query processing.

2 Related Work

Lu et al. [1] propose a parallel system to process query patterns with graph traversal approach and XML data is stored in relational databases with respect to DOM tree properties of arcs, vertices, labels, and values. They employ vertical and horizontal partitioning in relational databases directly, but do not explain clearly their data distribution strategy.

Tang et al. [3] introduce query rewriting and hash join operation with index structures in answering query patterns in its parallel system. The authors utilize parallel DBMS to store XML data. Their XML data partitioning strategy is to form sub-trees that will be maintained in different sites. The cost-based distribution model basically considers inherent structures of XML data, query expressions and index structures.

We previously proposed a structural join approach for processing query patterns in shared-nothing parallel system [4]. Path approach is used to map XML data into relational databases where XML data is stored in node table describing properties of node id, path id, node number, node type, value and in path table denoting path id and path expression. To achieve load balance, vertical and horizontal partitioning based on schema graph decomposition is carried out and cost-based allocation of partitions with respect to costs of some basic operators to execute parallel structural joins is studied thoroughly.

3 The Proposed Scheme

3.1 Multidimensional Data Construction

To construct a multidimensional data model for generating XML data partition, we outline three main steps in gathering statistics of XML data: (i) traversing XML documents, (ii) fetching queries from logs, and (iii) constructing the multidimensional data. Our system represents the multidimensional data using relational database system.

In the first step, every XML document is fed into the system and it results a document dimension containing distinct XML document ids and a tag dimension containing distinct tags in the entire set of XML documents. Based on these two dimensions, a 2-dimensional (2-D) structure that maps those two dimensions along with the number of tag occurrences as the mapping value is composed. Meanwhile, 3-tuple representation of every tag occurrence as described in [2] is enumerated and stored in a related stream of nodes. Subsequently, for every query twig pattern in the log history, its query id and its frequency of occurrences contributes to a query dimension. To cope with our partition scheme that due to cost imbalance a (twig) query might be decomposed into a set of query paths (root to leaf paths), the query dimension is refined in terms of a query path dimension. Every query path consisting of nodes associated with the tag dimension is extracted to form a 2-D structure mapping between the query path dimension and the tag dimension. Finally, the multidi-

mensional data is built thorough joining the two 2-D structures with the join key of tag values and ensuring that a set of query paths belonged to a query is fully mapped into an XML document. An instance of the multidimensional data has a meaning that a tag node of a query path that occurs in an XML document contributes a cost of query processing. Figure 1 illustrates the overview of the proposed scheme.

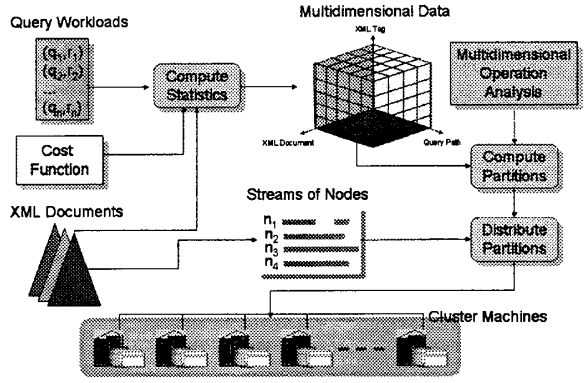

図 1: An overview of the proposed scheme.

3.2 Cost Model

Our cost model takes computation C_{comp} and communication C_{comm} complexities measured in terms of the number of node occurrences as the unit. In the worst case, a query twig pattern may be decomposed into query path patterns; thus, we consider the cost of a query path pattern as the foundation of computing the cost of a query twig pattern.

$$C(q_i) = f_q(\alpha C_{comm} + (1-\alpha)\ C_{comp}) \quad (1)$$

$$C(q_i) = f_q((\alpha + \alpha\gamma + \gamma)\textstyle\sum_{n \in inputs} |n|) \quad (2)$$

Formula (1), (2) express the cost of individual query path patterns, where f_q is the probability of query occurrence in the system and α ($\alpha \leq 1$) is the coefficient for weighting C_{comp} and C_{comm}. In the holistic twig joins [2] , basically C_{comp} is linear in the sum of sizes of the input lists and output lists. The size of output lists is estimated to a fraction γ of the size of input lists, where fraction γ value ($0 \leq \gamma \leq 1$) can be derived from statistics. Meanwhile, C_{comm} estimates the size of output lists ($\gamma|inputs|$) sent by a cluster machine to the coordinator. Therefore, the cost of a query is the sum of its query path costs.

3.3 Multidimensional Analysis

To perform XML data partition we provide some operations on the multidimensional data model. Since we have numerous XML documents in the system, in the first step we need to group those documents as our initial partitions. A map of XML document and XML tag dimensions is extracted from the data model as a 2-D projection to perform document clustering based on XML tag similarity. In this case, we select an agglomerative hierarchical clustering to produce clusters regarded as partitions, which contain groups of XML documents, including their associated costs. In general, the resulted costs of partitions are still considerably high, so that partitions need to be further refined. By analyzing another 2-D projection of query and XML tag dimensions in the data model, we perform query clustering according to tag similarity only for partitions having considerably high costs. In order to do this second clustering, a drill-up operation on query path dimension to yield query dimension is performed in advance and the similarity measure in terms of normalized costs is computed simultaneously.

Before conducting the actual XML data distribution to cluster machines, we simulate the distribution of partitions using greedy approach with the objective of minimizing the overall cost variance among the entire machines. As the result, we could identify cluster machines having overloaded costs; thus, further partition refinement is required to perform.

Additionally, to generate more refined partitions we consider a document-query mapping in the multidimensional data model to execute a query split operation on document if a query is mapped into several XML documents. Another approach of taking a query-tag mapping in the data model is to execute another query split operation on query paths if a single query has such a high cost.

4 Conclusions

In this paper, we propose a novel scheme of XML data partitioning for parallel holistic twig join processing. We focus on building the multidimensional data model and performing the multidimensional analysis to produce XML data partitions. In the future, we plan to study an adaptation of load balance changes due to new XML document addition into the system.

Acknowledgments

This study has been partially supported by MEXT (#19024006), Grant-in-Aid for Young Scientists (B) (#19700083) and CREST of JST (Japan Science and Technology Agency).

参考文献

[1] K. Lu, Y. Zhu, W. Sun, S. Lin: Parallel Processing XML Documents. Proc. Int'l Database Engineering and Applications Symposium (IDEAS), 2002.

[2] N. Bruno, N. Koudas, D. Srivastava: Holistic Twig Joins: Optimal XML Pattern Matching. Proc. ACM SIGMOD, 2002.

[3] N. Tang, G. Wang, J. Xu Yu, K. Wong, G. Yu: WIN: An Efficient Data Placement Strategy for Parallel XML Databases. Proc. Int'l Conf. on Parallel and Distributed Systems (ICPADS), 2005.

[4] T. Amagasa, K. Kido, H. Kitagawa: Querying Data Using PC Cluster System. Int'l Workshop on XANTEC'07 in conjunction with Database and Expert Systems Application (DEXA'07), 2007.