ProDA: A Suite of Web-Services for Progressive Data Analysis*

Mehrdad Jahangiri
University of Southern California
Department of Computer Science
Los Angeles, California, 90089-0781
jahangir@usc.edu

Cyrus Shahabi
University of Southern California
Department of Computer Science
Los Angeles, California, 90089-0781
shahabi@usc.edu

ABSTRACT

Online Scientific Applications (OSA) require statistical analysis of large multidimensional datasets. Towards this end, we have designed and developed a data storage and retrieval system, called ProDA, which deploys wavelet transform and provides fast approximate answers with progressively increasing accuracy in support of the OSA queries. ProDA employs a standard web-service infrastructure to enable remote users to interact with their data. These web-services enable wavelet transformation of large multidimensional datasets as well as inserting, updating, and exact, approximate and progressive querying of these datasets in the wavelet domain. We demonstrate the features of ProDA on a massive atmospheric dataset provided to us by NASA/JPL.

1. INTRODUCTION

Following the constant technological advancements that provide more processing power and storage capacity, scientific applications have emerged as a new field of interest for the database community. Many Online Scientific Applications (OSA) require continuous interaction with datasets of multidimensional nature, mainly for performing statistical analysis. Currently, most of the OSA computations occur at the application side increasing both the data transfer and the client-side processing, while not exploiting process sharing and result caching at the server side.

To shift some of the application side processing to the storage server, Online Scientific Applications can seriously benefit from the ongoing research in the OLAP systems and the precalculation of aggregate functions for multidimensional datasets. However, there are some main differences between OLAP and OSA. Generally speaking OSA needs to perform more complex statistical queries on very large multidimensional datasets; thus, a number of multivariate statistical methods (e.g., calculation of covariance or kurtosis) must be supported. On top of that, the desired accuracy varies per application, user and/or dataset and it can well be traded-off for faster response time. It is therefore important that all this functionality is inherently supported by the storage subsystem, on top of which OSAs can be easily built. We believe that our methodologies described in this paper can contribute towards this end.

We have designed and developed a system, named ProDA for *Progressive Data Analysis*, which utilizes the wavelet decomposition of multidimensional data to achieve progressiveness by retrieving data based on the significance of both query and data values. The use of the wavelet decomposition is justified by the well-known fact that the query cost is reduced from being a function of the query size to a function of the logarithm of the data size, which is a major benefit especially for large range queries. Wavelet multi-resolution property elevates this by re-distributing the energy of data and retrieving based on the data significance for progressive query processing. Our system, as described in this paper, is effectively built on our previous work in query estimation using wavelets [2, 4, 5] where wavelets are used as a tool for approximate or progressive query evaluation. Using these techniques, ProDA supports any high order polynomial range-aggregate query (e.g., variance, covariance, correlation, etc). We demonstrate all these features of ProDA on a massive atmospheric dataset and a reservoir simulation dataset respectively provided to us by NASA/JPL and Chevron-Texaco.

The remainder of this paper is organized as follows. In Section 2 we discuss the research underlying ProDA. We explain the architecture of ProDA in Section 3. Finally, Section 4 concludes the paper.

2. RESEARCH BACKGROUND

The Discrete Wavelet Transformation provides a multiscale decomposition of data by creating "rough" and "smooth" views of the data at different resolutions. In the simple case of the Haar wavelets, the "smooth" view is comprised of averages, whereas the "rough" view is comprised of details, or differences. At each resolution, termed level of decomposition or scale, the averages and details are constructed by pairwise averaging and differencing of the averages of the previous level. We refer the reader to [**?**] for further discus-

*This research has been funded in part by NSF grants EEC-9529152 (IMSC ERC) and IIS-0238560 (PECASE), unrestricted cash gifts from Microsoft, an on-going collaboration under NASA's GENESIS-II REASON project and partly funded by the Center of Excellence for Research and Academic Training on Interactive Smart Oilfield Technologies (CiSoft); CiSoft is a joint University of Southern California - ChevronTexaco initiative.

SIGMOD 2005 June 14-16, 2005, Baltimore, Maryland, USA

sion about Wavelets.

Wavelets are often thought of as a data approximation tool, and have been used this way for approximate range query answering. The efficacy of this approach is highly data dependent; it only works when the data have a concise wavelet approximation. Furthermore the wavelet approximation is difficult to maintain. To avoid these problems, we use wavelets to approximate incoming queries rather than the underlying data. Note that the data set is still transformed using wavelet; however, it is not approximated since we keep all the coefficients. We are able to obtain accurate, data-independent query approximations after a small number of I/Os by using the largest query wavelet coefficients first. This approach naturally leads to a progressive algorithm. The details of our techniques can be found in [2, 3, 4, 5]. ProDA is an implementation of these techniques and supports the following features:

- ProDA treats all dimensions, including measure dimensions, symmetrically and supports range-aggregate queries where the measure can be any polynomial in the data dimensions (not only COUNT, SUM and AVERAGE, but also VARIANCE, COVARIANCE and more). All computations are performed entirely in the wavelet domain.

- ProDA uses the lazy wavelet transform to achieve query and update cost comparable to the best-known exact techniques.

- By using the most important query wavelet coefficients first, ProDA provides excellent approximate results and guaranteed error bounds with very little I/O and computational overhead, reaching low relative error far more quickly than analogous data compression methods.

3. SYSTEM ARCHITECTURE

ProDA has a 3-tier architecture. The bottom tier is the data storage tier, which is optionally implemented either as a DBMS (for easy data management) or as a custom file-system (to achieve efficiency). The middle-tier is a query processing module comprising a set of web-services developed in the Microsoft .NET framework. Finally, the top tier consists of a web-accessible Graphical User Interface (GUI) implemented in C#. We encourage the reader to visit the ProDA website at [1].

3.1 Bottom tier

For this tier, we can either use any commercial DBMS or implement our own customized file-system. Here, we describe only the first option due to the space limitation.

We chose Oracle as our relational database management system for data storage and we modelled our multidimensional data cubes using the star schema. In the star schema design, we used a fact table to store all data wavelet coefficients and a dimension table for each cube dimension to store dimension values. Moreover, we stored properties of all dimensions (such as dimension sizes, dimension titles, ...) in a separate table. Finally, we maintained a cube directory listing all cube names stored in the database. Figure 1 illustrates this schema.

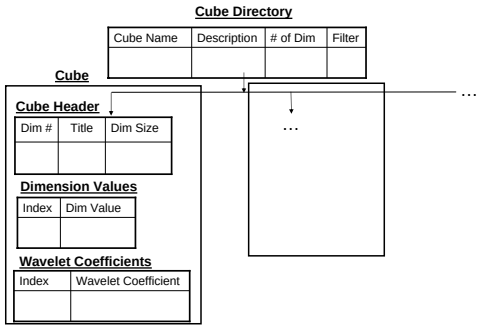

Figure 1: Database Architecture

3.2 Middle tier

ProDA provides a rich set of web-serivces listed in [1]. This tier consists of four sets of web-services as follows:

- **General:** This set of web-services provides easy interaction with wavelet decomposition for non-expert users. For example, we can ask for the list of supported wavelet filters and properties of each filter. By invoking these web-services, we can wavelet transform or reverse-transform a multidimensional datacube given a specific filter name.

- **Cube Maintenance:** By having maintenance privileges, we can insert, delete and update cubes through this set of web-services. We have also provided an appending feature which can preserve the best synopsis in case of facing with a data stream scenario.

- **Cube Access:** We can select a cube and access its content by invoking these web-services. We can retrieve any cube information through this set of web-services. Also, we can ask for the list of available cubes and their metadata.

- **Query:** This set includes cursor functions, predefined aggregate functions and polynomial aggregate functions. Cursor functions are designed to track the progress of the query operations. Many statistical range-aggregate queries are predefined and implemented in ProDA such as count, sum, average, variance, covariance, and correlation. However, we can form any arbitrary polynomial range-aggregate query by using two basic web-services. These two web-services are *PushTerm* and *PushOperator*, using which we can form any polynomial query and submit its post-order to the system.

Example 1. Consider the case we need to re-implement the variance function using *PushTerm* and *PushOperator*. Variance is defined as following:

$$Var(x) = \frac{\sum x_i^2 - (\sum x_i)^2}{n}$$

The post order representation of this function is $\sum x_i^2$, $\sum x_i$, ^{2}, $-$, n and $/$. Therefore, we can submit our polynomial query with 7 calls as shown in Figure 2.

Example 2. Consider the case we need to implement regression coefficient, defined as $Cov(x, y)/\sqrt{Var(x)}$, using the same operators.

```
// Assume dimId(x)=1
    PushTerm(1,2);          // ∑ x_i^2
    PushTerm(1,1);          // ∑ x_i
    PushOperator('**');     // ^2
    PushOperator('-');      // −
    PushTerm(1,0);          // n
    PushOperator('/');      // /
    SubmitQuery();          // submit
```

Figure 2: Re-implementing a Polynomial Query

We can form this function with 4 calls as shown in Figure 3, knowing the fact that $Var()$ and $Cov()$ functions have already been defined in ProDA.

```
// Assume dimId(x)=1 & dimId(y)=2
    Cov(1,2);               // cov(x, y)
    Var(1);                 // var(x)
    PushOperator('//');     // √
    PushOperator('/');      // /
    SubmitQuery();          // submit
```

Figure 3: Implementing a new Polynomial Query

3.3 Top tier

A client is provided for the user to issue various range-aggregate queries on a selection of attributes. Once a user submits the query, the client can display the result and its relative error, and update them periodically (e.g., every half a second). A user can interrupt the progression at any time.

Figure 4 depicts the C# code of a sample ProDA client in five steps. First, we create an instance of our services and store a session state for further use. Next, we ask for the list of available cubes. Later, we select a cube and display its properties, e.g., we display titles of all dimensions. Next, we specify a range and submit a query. Finally, we utilize our cursor services to obtain the query result progressively.

```
// Creating an instance and storing session state
ProDAWebServices pws=new ProDAWebServices();
pws.Credentials = System.Net.CredentialCache.DefaultCredentials;
pws.CookieContainer = new CookieContainer();
// Listing all cube names
string []dbnames=pws.GetAllCubeNames();
for(int i=0;i¡dbnames.Length;i++)
    Console.WriteLine(dbnames[i]);
// Selecting a cube and printing some properties
pws.SelectDB("...");
int n=pws.GetDimN();
for(int i=0;i¡n;i++)
    Console.WriteLine("Dim"+i+"("+pws.GetDimTitle(i)+")");
// defining a range and submitting a query
int []rStart={...};
int []rEnd ={...};
pws.SetRange(rStart,rEnd);
pws.Varariance(1);
pws.SubmitQuery();
// Asking for result progressively
while(pws.HasMore())
    Console.Write("Result="+pws.Advance(100));
pws.CloseConnection();
```

Figure 4: A sample code for a ProDA client for progressive querying (in C#)

3.4 Customized Client

We have developed a number of customized GUIs to provide a friendly application-specific interface for a various application domains. In this paper we only discuss the GUI we have built for our NASA/JPL application (see Figure 5). Here, we are interacting with a particular dataset, namely atmospheric temperature data provided by NASA/JPL. This dataset is a four-dimensional datacube including latitude, longitude, altitude and time as dimension attributes, and temperature as the measure attribute. The corresponding domain size for the above dimensions are 64, 128, 16, and 8192, resulting in a datacube with more than 1 billion cells. The user can specify latitude, longitude, altitude and time ranges, and query the temperature. The current queries supported by our GUI are: count, sum, average, covariance, variance, standard deviation, and correlation.

4. CONCLUSION

ProDA enables OSA to interact with massive multidimensional datasets. Standard methods that rely on massive precomputation of aggregates are very expensive to update, and relational methods that are easy to update often have very slow query response. However, we have employed wavelets to support exact, approximate and progressive statistical range-aggregate queries on large multidimensional data-sets while keeping update cost relatively low. We pushed most of the application side processing to our storage system to enhance the overall efficiency. ProDA supports any arbitrary polynomial query and provides progressive query answering.

Figure 5: Customized GUI to query atmospheric temperature dataset

5. REFERENCES

[1] ProDA: http://infolab.usc.edu/projects/proda/.

[2] M. Jahangiri, D. Sacharidis, and C. Shahabi. SHIFTSPLIT: I/O Efficient Maintenance of Wavelet-Transformed Multidimensional Data. In *Proceedings of ACM SIGMOD*, 2005.

[3] R. Schmidt and C. Shahabi. How to evaluate multiple range-sum queries progressively. In *Proceedings of ACM PODS*, pages 3–5.

[4] R. Schmidt and C. Shahabi. Propolyne: A fast wavelet-based technique for progressive evaluation of polynomial range-sum queries. In *Proceedings of EDBT*, 2002.

[5] C. Shahabi, M. Jahangiri, and D. Sacharidis. Hybrid Query and Data Ordering for Fast and Progressive Range-Aggregate Query Answering. *International Journal of Data Warehousing and Mining*, 1(2):49–69, April-June 2005.