Fast Approximate Evaluation of OLAP Queries for Integrated Statistical Data*

Jose Luis Ambite[1], Cyrus Shahabi[2], Rolfe R. Schmidt[2], and Andrew Philpot[1]
Digital Government Research Center
Information Sciences Institute[1] and Computer Science Department[2]
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292, USA

Abstract

We have developed a mediator architecture that integrates statistical information about energy products from several government agencies, such as the Bureau of Labor Statistics, the Energy Information Administration, and the California Energy Commission. Our architecture has a dual mode of operation. First, our system can retrieve live data from databases and web sources from these agencies. This allows the users to obtain completely up-to-date data. However, for complex analytical queries that typically require large amounts of data and processing, live access does not offer the level of interactivity that some users require. Second, our system can warehouse the information from the data sources to allow for complex analytical queries to be executed much more efficiently. However, the data would be only as recent as the last update to the data warehouse. In this paper we describe the architecture and focus on how to perform analytical queries against the data warehouse very efficiently. We present results using a fast wavelet-based technique for progressive evaluation of range-sum queries. This technique allows for returning an approximate result to the query very efficiently and for fast convergence to the exact result. We envision users exploring many complex queries using the very fast approximate results as guidance and only obtaining the exact results for those queries that are deemed of interest. We present experimental results showing the efficiency of both approximate and exact queries.

1 Introduction

Governments produce a wealth of information for the both the general public and analysts. The different government agencies make this information available in a variety of formats: text publications, databases, web sites with static data or with form-based interfaces that allow a limited form of data querying, CD-ROM applications, etc. Of particular importance is statistical data collected and published as part of the mandate of many agencies, since this data is a high-quality reference for analysis of many sectors of economic activity.

Finding the data relevant to a given application domain often requires the user to gather information across multiple agencies. The informational needs of general users and analysts would be greatly facilitated if there was a single point of access and a uniform access mechanism to all the required data without the

*This research was supported by the National Science Foundation's Digital Government Program under contract EIA-9876739. Shahabi and Schmidt were supported in part by NSF grants EEC-9529152 (IMSC ERC) and ITR-0082826, NASA/JPL contract nr. 961518, DARPA and USAF under agreement nr. F30602-99-1-0524, and unrestricted cash/equipment gifts from NCR, IBM, Intel and SUN.

user needing to worry about location, format, and terminologies of different agencies. Moreover, integrated access to data is an important and necessary first step, but often analysts need to evaluate complex queries over this data while still obtaining fast responses.

In the Energy Data Collection (EDC) at the Digital Government Research Center (DGRC) we are addressing these types of information integration problems. The Energy Data Collection project is building an integration system for accessing and disseminating energy data from sources such as the Bureau of Labor Statistics (BLS), the Census Bureau, the Department of Energy's Energy Information Administration (EIA), and the California Energy Commission (CEC) [Ambite et al., 2001].

In this paper, we build on our previous work in the EDC project and we propose an architecture for fast evaluation of OLAP (On-Line Analytical Processing) queries on our integrated data. In the first place we construct a data warehouse of energy data integrated from multiple agencies. Then, we describe a very efficient method of processing range-sum queries based on wavelet techniques [Schmidt and Shahabi, 2001b, Schmidt and Shahabi, 2001a]. Range-sum queries are some of the most useful OLAP queries. Our wavelet methods provide a very fast approximate answer that converges rapidly towards the exact answer. Even our computation of an exact answer is competitive with the best approaches for this type of queries.

The rest of the paper is organized as follows. First, we describe our integration approach and how it supports both live query access and materialization in a local data warehouse. Then, we describe our progressive OLAP query evaluation based on wavelets and experimental results on our energy domain. Finally, we conclude with a discussion of our approach and future work.

2 Information Integration

In the first phase of the EDC project we developed a mediator architecture that integrates time-series data from several web sites and databases from agencies such as the Bureau of Labor Statistics (BLS), the Census Bureau, and the Energy Information Administration (EIA). First, we briefly present our mediator architecture and the modeling of time-series data. We refer the reader to [Ambite et al., 2001] for details. Then, we describe how we extend this architecture by incorporating a local data warehouse to allow for more efficient processing of OLAP queries.

The architecture of the EDC system appears in Figure 1. The are three principal components: the database manager and access planner, the overarching ontology in which terms are defined and standardized, and the interface. The ontology has descriptions of all the concept of interest in our domain in addition to 90,000 common concepts [Knight and Luk, 1994]. The query planner (SIMS) [Arens et al., 1996, Ambite and Knoblock, 2000] answers a user query by considering these concept descriptions and finding which sources have the relevant data. The interface provides an intuitive mechanism for the users to specify queries against the system.

Each time series is described in the ontology as a conjunction of hierarchical dimensions, such as product type (e.g., unleaded gasoline, premium gasoline), property measured (e.g., price, volume), area of the measure (e.g., USA, California), unit of measure, etc. A fragment of the ontology showing these dimensions appears in Figure 2. Each of the time series in the sources is described by using specific values for each of the hierarchical dimensions. For example, a particular source may be described as providing the monthly prices (based on the consumer price index) of premium unleaded gasoline for the state of California. The dimensions can be seen as metadata that describes the series. The actual data is modeled as a set of measurements (i.e., date and value pairs). The domain model also describes whether a source has footnotes for some of the data.

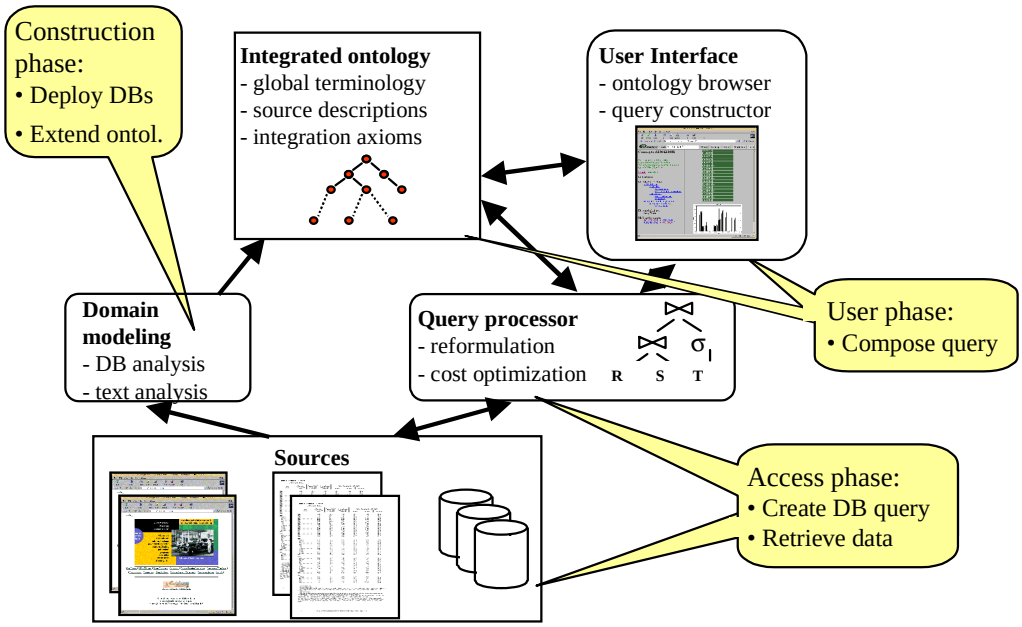


Figure 1: Mediator Architecture

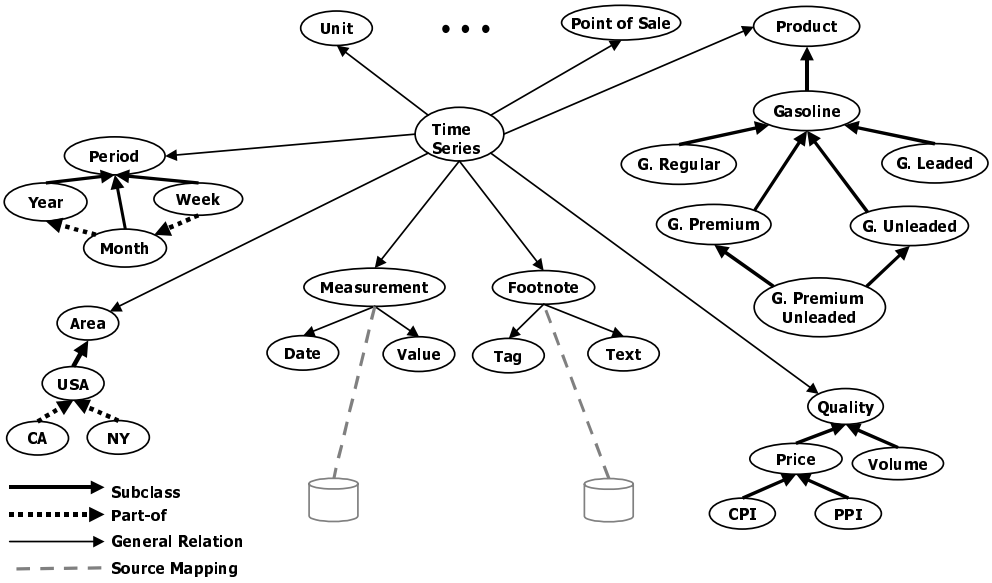
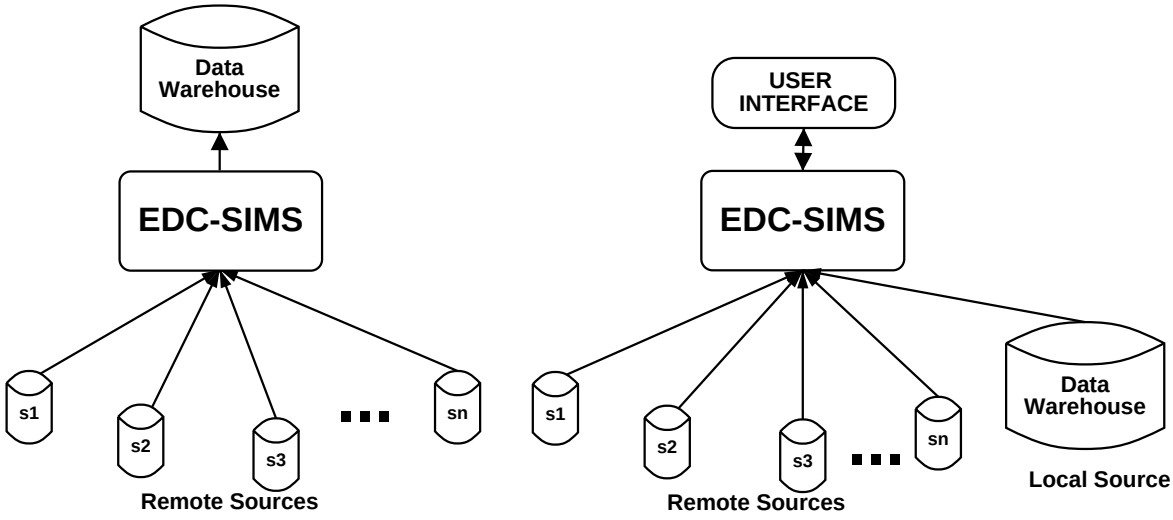


Figure 2: Dimensions in Time Series Ontology

Our architecture has a dual mode of operation. First, our system can retrieve live data from databases and web sources. This allows the users to obtain completely up-to-date data. However, for complex analytical queries that typically require large amounts of data and processing, live access does not offer the level of interactivity that some users require. Second, our system can warehouse the information from the data sources to allow for complex analytical queries to be executed much more efficiently. However, the data would be only as recent as the last update to the data warehouse. This extended architecture is shown in Figure 3. Figure 3(a) shows the process of loading the data warehouse. Once we have the sources modeled in our ontology we can retrieve all the available time-series and store them in a local data warehouse normalized under our common schema. On this local warehouse we can evaluate queries much more efficiently as we describe in the next section. Figure 3(b) shows how we propose to use the data warehouse. The user interacts with a friendly interface that passes formal queries (e.g. SQL) to the query planner (SIMS). Then the query planner analyzes the query and decides whether to retrieve the data live or to use the local data warehouse. In this paper we do not address the reasoning task that allows the mediator to decide whether to use the warehouse or the live sources. In next section we describe how the mediator efficiently evaluates complex analytical queries against the data warehouse.



(a) Loading the Data Warehouse (b) Using the Data Warehouse

Figure 3: Supporting our integration system with a Data Warehouse

3 POLAP: Fast Wavelet-based Progressive Evaluation of OLAP Queries

Range sum queries are one of the most basic types of decision support query, but even the best proposed techniques for their evaluation scale poorly with the dimension of the data domain. [Schmidt and Shahabi, 2001a, Schmidt and Shahabi, 2001b] introduced POLAP a new algorithm that produces exact range sum query results as efficiently as the best known methods, but also produces accurate estimates of the query result long before the exact computation is complete. This combination allows us to deliver progressive query evaluation: we provide a quick approximate query response that becomes increasingly accurate as the computation progresses. In this paper we apply POLAP to statistical data integrated from multi-

ple agencies. This section presents an introduction to the main ideas of POLAP. We refer the reader to [Schmidt and Shahabi, 2001a, Schmidt and Shahabi, 2001b] for a more detailed technical description. In the next section we show some experimental results for time-series data from our energy domain.

POLAP, our progressive algorithm for range-sum query evaluation can be used as an exact algorithm, and as such it has query I/O complexity of $O(((\log\ N)/d)^d)$. One variation of the algorithm has storage complexity of $O(\log\ N)$ during computation, allowing much larger problems to be executed in memory. Furthermore, the cost of updating the database using our technique is $O(((\log\ N)/d)^d)$. To our knowledge there are no existing algorithms that have lower query complexity without having higher update cost. POLAP can also be used to produce accurate approximate query results by terminating the exact computation before it is complete. We demonstrate with an implementation that these approximate query results become meaningful after a small fraction of the I/O is complete. Used as a progressive algorithm, POLAP can provide users with these increasingly accurate query estimates during the process of exact query evaluation.

Range Queries. For simplicity we restrict our attention to data that lie in a d-dimensional data cube D where each dimension has 2^{j_*} points. The entire domain has $N = 2^{dj_*}$ points.

Definition 1 *A* Range *in D is a rectangular region of the form $R = \prod_{i=0}^{d-1}[l_i, h_i]$ where $l_i \leq h_i$.*

An additive range query is usually defined as the sum of some measure attribute over the points in the database that lie in the range. A simple example in our gasoline domain would be the sum of gasoline volumes consumed over all states and grades. These sums can be written as integrals over the range of the product of a measure function and the data density function [Shanmugasundaram et al., 1999]. This is the formulation we will use to make the following definition.

Definition 2 *Given a function $f : D \rightarrow \mathbb{C}$ (the measure attribute), a range $R \subset D$, and a database of points in D with density function $\Delta : D \rightarrow \mathbb{R}^+$, the range query $Q(R, f, \Delta)$ is defined as*

$$Q(R, f, \Delta) = \sum_{x \in D} f(x)\chi_R(x)\Delta(x) \tag{1}$$

where χ_R denotes the characteristic function of the set R. The function $f\chi_R$ is referred to as the range function *for the query.*

The Core of POLAP. Definition 2 shows the association between a range-sum query and a range function. The fundamental idea behind POLAP is that we can express these range functions as a sum of basic component functions. Each of the component functions can be thought of as a sub-query, and we choose a decomposition and a data storage format so that each of these sub-queries can be computed in constant time. Once the query is decomposed, we sort the sub-queries in order of importance for the final result. Evaluating the most important sub-queries first gives a progressively more accurate estimate of the eventual exact result. Thus, our algorithm can be used as an exact, approximate, or progressive range-sum query evaluator.

While simple at a high level, there are some fundamental challenges to this approach. In order to be viable, we need to find a good decomposition of range functions. The decomposition must have a uniformly small number of terms for all range functions, so that the query complexity does not depend on the nature of the range itself. We also must have a way to evaluate the sub-queries quickly. Fortunately, a wavelet-based approach satisfies both requirements.

A wavelet decomposition of a range function gives us sub-queries that can be evaluated easily. The wavelet transforms we use decompose a function into a sum of orthogonal wavelets:

$$f(x)\chi_R(x) = \sum_{\mathbf{j},\mathbf{k}} \nu_{\mathbf{j},\mathbf{k}}(f\chi_R)\psi_{\mathbf{j},\mathbf{k}}(x)$$

This allows us to rewrite the sum in Definition 2 as

$$Q(R, f, \Delta) = \sum \nu_{\mathbf{j},\mathbf{k}}(f\chi_R)\nu_{\mathbf{j},\mathbf{k}}(\Delta) \tag{2}$$

Moreover, the decomposition of range functions in a basis of wavelets can be performed very efficiently. If we store the values $\nu_{\mathbf{j},\mathbf{k}}(\Delta)$, we can evaluate the sub-queries $\nu_{\mathbf{j},\mathbf{k}}(f\chi_R)\nu_{\mathbf{j},\mathbf{k}}(\Delta)$ as quickly as we can compute the wavelet transform of $f\chi_R$.

Finally, our approach would not be practical if there were too many sub-queries in our decomposition. Fortunately, most of the wavelet coefficients of typical range functions $f\chi_R$ turn out to be zero and the few remaining coefficients can be computed quickly. The fact that wavelets decompose range-sum queries into a sum of a small number of easy to compute sub-queries gives us exactly what we need to carry out our high level progressive query evaluation strategy.

4 Performance Evaluation

We have implemented POLAP in C++ using Db4 wavelets. This implementation has been tested on real data from our gasoline domain. While these results appear to compare favorably with other approximate OLAP algorithms [Vitter et al., 1998, Shanmugasundaram et al., 1999], we do not attempt a direct comparison. The purpose of this demonstration is to show how quickly POLAP converges to the exact result, and how this convergence is affected by the sparseness of the data, the size of the query relative to the domain, and the size of the domain itself. Throughout this section, we use the number of accesses to the POLAP database as a proxy for the number of I/Os.

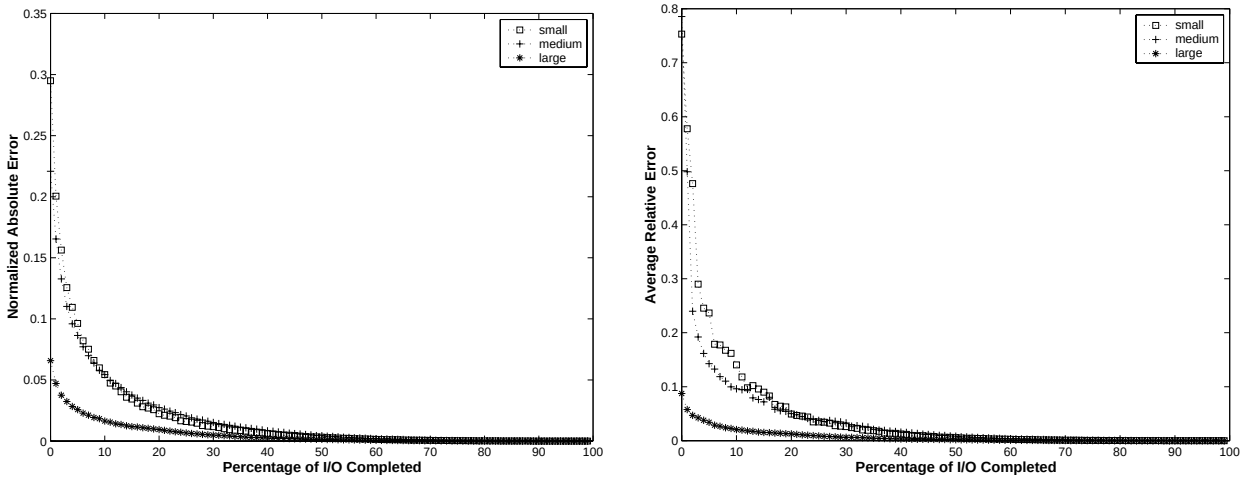
Error Measures. We report query error using both average relative error and a normalized absolute error. We attempted to use the absolute error normalization of [Vitter et al., 1998] which divides each error by the largest value of the partial sum cube, but found that our normalized errors looked exceptionally good. Hence we propose a more conservative measure: given a class of queries, we divide the average absolute error by the average magnitude of all nonzero query results. This measure provides us with an estimate of how well we can distinguish queries that will progress to zero from those that have a nonzero exact answer.

Query Generation. We have tested POLAP for a variety of range shapes and sizes. Here we present detailed results for three representative query classes. We selected 6 dimensions on which to query, giving a queriable datacube with 2.35 million cells and a sparseness of 0.7%. With the dimensions specified, we execute COUNT queries for three classes of ranges: small, medium, and large. Small ranges have edges of a small constant size in each dimension. The medium ranges have side lengths proportional to the square root of the corresponding dimension size. The large ranges have side lengths equal to one-half of the corresponding dimension size. These range sizes were chosen so that the wavelet transform of the small ranges would fall primarily in the high resolution levels, the wavelet transform of the large ranges will fall primarily in the low resolution levels, and the wavelet transform of the medium ranges will be concentrated

at the middle resolution levels. We also restrict our attention to queries returning non-zero results. This restriction is reasonable: interactive OLAP users will issue drill-down queries on interesting regions. If a region contains no data, there is no need for follow-up queries. One thousand queries for each range size were generated uniformly throughout the data domain. Error statistics were accumulated over these query evaluations to produce the results reported here.

Results. Figure 4 demonstrates the rate at which POLAP converges to the exact result for the petroleum data set. Figure 4(a) shows the normalized absolute error of the POLAP query result plotted versus the percentage of total I/O completed. Each curve in the graph represents a different range size, results are displayed for small, medium, and large ranges. Here the results are striking: after 3% of the I/O is complete, the average absolute error for each range type is less than 20% of the average nonzero query result. After 6% of the I/O is complete, the average absolute error is less than 10% af the average query result. We also observe that POLAP converges more quickly for large ranges than it does for small ranges.

Figure 4(b) is similar to Figure 4(a), but reports average relative error instead of absolute error. The relative error follows very similar trends. After 7% of the I/O is complete, the average relative error for all range types is less than 20%. After 14% of the I/O is complete this improves to less than 10%. For large ranges, the average relative error was less than 10% after only 1% of the I/O was complete.



(a) Absolute Error Vs. I/O

(b) Relative Error Vs. I/O

Figure 4: Query Evaluation Progress in POLAP for our gasoline domain

5 Discussion

We have described an integration architecture that can operate both in a virtual mode and in a materialized mode. When our system works as a mediator the integration is virtual in the sense that the actual data reside in the remote sources. The data become integrated only through the models and query processing in the mediator. Our system can also materialize all the available data in a local warehouse. Once the data is in the warehouse, the system can process complex analytical queries very efficiently even for very massive data

sets by using POLAP our wavelet-based query evaluation algorithm that can produce both exact answers competitively with the best algorithms or very fast approximate answers.

In the future we plan to explore the combination of the virtual and materialized approaches as was shown in Figure 3(b). The mediator may answer a user query using both the materialized warehouse and the remote sources so that the evaluation time is optimized. Another area of future research is integrating data sets at multiple granularities taking advantage of the summarization properties of wavelets. Recall that in our description of the time-series we used hierarchical attributes. For example, one series may be a monthly measurement but another may be a yearly measurement. If we wanted to obtain data at a yearly granularity we could aggregate the monthly data. But what if we wanted the data at a monthly granularity? We plan to use the properties of wavelets to infer the lower granularity data according to the distribution of the data so that we can do the best approximation possible and evaluate queries that would have been impossible to answer without our integration system.

References

[Ambite et al., 2001] Ambite, J. L., Arens, Y., Hovy, E., Philpot, A., Gravano, L., Hatzivassiloglou, V., and Klavans, J. (2001). Simplifying data access: The energy data collection (EDC) project. *IEEE Computer, Special Issue on Digital Government*, 34(2).

[Ambite and Knoblock, 2000] Ambite, J. L. and Knoblock, C. A. (2000). Flexible and scalable cost-based query planning in mediators: A transformational approach. *Artificial Intelligence Journal*, 118(1-2):115–161.

[Arens et al., 1996] Arens, Y., Knoblock, C. A., and Shen, W.-M. (1996). Query reformulation for dynamic information integration. *Journal of Intelligent Information Systems, Special Issue on Intelligent Information Integration*, 6(2/3):99–130.

[Knight and Luk, 1994] Knight, K. and Luk, S. K. (1994). Building a large-scale knowl-edge base for machine translation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 773–778, Seattle, WA.

[Schmidt and Shahabi, 2001a] Schmidt, R. R. and Shahabi, C. (2001a). Polap: A fast wavelet-based technique for progressive evaluation of olap queries. Submitted.

[Schmidt and Shahabi, 2001b] Schmidt, R. R. and Shahabi, C. (2001b). Wavelet based density estimators for modeling OLAP data sets. In *Third Workshop on Mining Scientific Datasets, in conjunction with First SIAM Int'l Conference on Data Mining*.

[Shanmugasundaram et al., 1999] Shanmugasundaram, J., Fayyad, U., and Bradley, P. (1999). Compressed data cubes for OLAP aggregate query approximation on continuous dimensions. In *Fifth ACM SIGKDD Inter. Conf. on Knowledge Discovery and Data Mining*.

[Vitter et al., 1998] Vitter, J. S., Wang, M., and Iyer, B. R. (1998). Data cube approximation and histograms via wavelets. In *Proc. of the 7th Int'l Conf. on Information and Knowledge Management*, pages 96–104. ACM.