Web Log Data Sparsity Analysis and Performance Evaluation for OLAP

Ji-Hyun Kim*, Hwan-Seung Yong*
*Department of Computer Science and Engineering
Ewha Womans University
11-1 Daehyun-dong, Seodaemun-gu, Seoul, 120-750, Korea
{pshike, hsyong}@ewha.ac.kr

Abstract: Many Internet businesses usually collect hundreds of megabytes of click-stream data everyday and want to analyze these data efficiently. But it is not easy to perform systematic analysis on such a huge amount of data. OLAP can be used for this purpose. It has to precalculate multidimensional summary results in order to get fast response. But as the number of dimensions and sparse cells increase, *data explosion* occurs seriously and the performance of OLAP decreases. In this paper, we present why this sparsity of web log data occurs and then what kinds of sparsity patterns are in the two and the three dimensions in OLAP. Based on sparsity patterns found we evaluate the performance of three OLAP systems (MS SQL 2000 Analysis Service, Oracle Express and MOLAP with the chunking method).

1. Introduction

Recently in competitive business environments, IT for CRM has been growing and developed rapidly. Typical applications are statistical analysis tools, on-line multidimensional analytical processing (OLAP) tools, and data mining algorithms (such neural networks, decision trees, and association rules). Business data that these applications analyze includes customer data, payroll data, inventory data, supplier data and competitive data, etc [1]. Specially, web click stream data among customer data can be easily obtained. But because of a tremendous of the customer click stream data, enterprises are laboring to answer even basic business questions, such as "which page is most popular". To use these data efficiently, they must set up the online analytical processing (OLAP) cube. But In this process, the sparsity appears and as a result, *data explosion* occurs in pre-calculation processing.

In this paper, we introduce why the web log data sparsity occurs and what kinds of sparsity patterns are generated in the two and the three dimensions in section 3. The data and the query model for evaluating the performance of three OLAP systems (MSSQL, Oracle, MOLAP with chunking method) are described in section 4. And we present the benchmark system and performance results in section5, 6 and conclude in section 7.

2. Related work

OLAP Products are classified into ROLAP (Relational OLAP), MOLAP (Multidimensional OLAP) and HOLAP (Hybrid OLAP) by storage structures. ROLAP systems by definition use relational tables as their data structure. This means that a cell in a logically multidimensional space is represented in the system as a tuple, with some attributes which describe corresponding data cell. So it takes much time and can have storage overheads to build a cube and to process OLAP operations. On the other hand, MOLAP systems store their data as multidimensional arrays. The data values are stored in fixed positions of arrays indexed by each dimension value so that the computation can be done efficiently. HOLAP systems are a combination of both ROLAP and MOLAP [2]. MOLAP computes faster than ROLAP but has to stores lots of sparse cells and sparse cells occur data explosion problem to decrease overall performance of OLAP system [3,4]. This becomes more serious as the number of dimension increases.

The existing methods for controlling the sparsity of OLAP are the composite dimension method of Oracle Express [5], the Sparse-Dense Split method of Hyperion EssBase [6] and Chunking Method [7].

3. Web log data sparsity analysis for OLAP

3.1 Why web log data is sparse

When we model web log data multi-dimensionally, it usually is sparse by several reasons. First, some attributes can have skewed distribution of values by their characteristic nature. For example, a ski resort site is usually crowded during late fall and winter season only. Then in a time dimension, there are many visitors for this duration but nobody for other seasons. Second, it occurs when a database design is changed as adding new attributes. The third reason is from system errors such as web server errors. Finally, it occurs by attributes which have meaning for a fixed period of time (for example, when these attributes are related with specific event). Among these reasons, the first one is the main cause of sparsity. In the next section, we explain three actual log data and the data model for the sparsity analysis.

3.1.1 Actual log data and data model

The type of web log data is based on the W3C extended format.

Before it is applied by the OLAP analysis, this data is preprocessed by the data cleaning, the user identification and the session identification module[8]. The first site delivers the world sports news and requires the registration. Detail information is shown in Table 1.

Table 1. Raw log data: first site

Duration	2000. 10. 28 ~ 2000.11. 4(a week)	
Data Size	Raw log data	2,513,975 records (163M)
	Preprocessed data	35,635 records (1.83M)

A data model combined with the preprocessed data and the customer data is shown in Figure 1. The second one is my laboratory's home page. This site does not require registration and as shown in Table 2, we used log data during fifteen days.

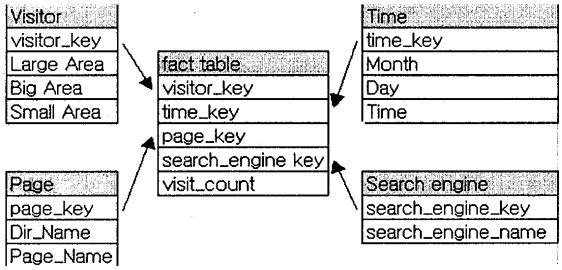

Figure 1. Data model: first site

Table 2. Raw log data: second site

Duration	2001. 03.01 ~ 2001.03. 15	
Data Size	Raw log data	198,591 records(120M)
	Preprocessed data	72,309 records(25.1M)

The data model is shown in Figure 2.

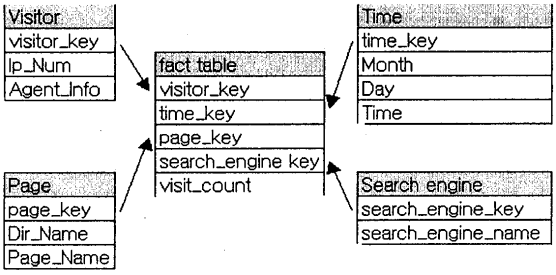

Figure 2. Data model: second/third site

The third site (Internet broadcast site) also doesn't require registration. The data model and site information are shown in Figure 2 and Table 3.

Table 3. Raw log data: third site

Duration	2001. 03. 28 ~ 2001.04. 03 (a week)	
Data Size	Raw log data	21,575 records(13.1M)
	Preprocessed data	19,739 records(4.72M)

3.1.2 Analyzing why the sparsity occurs in each dimension

As we have mentioned, web log data sparsity occurs mainly when attributes of each dimension are separated into dense and sparse ones and the number of the sparse attribute is more than one of the dense attribute. In this section, we explain why the sparsity occurs using three web log data. The visitor dimension of the first site is shown in Figure 3. We can find that people of Seoul and some regions visit frequently this site but others not.

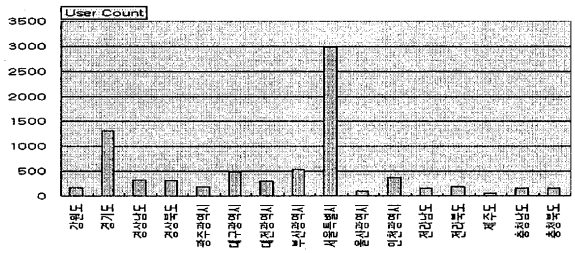

Figure 3. Visitor dimension

Second, the page dimension is also divided into interesting pages and non-interesting pages.

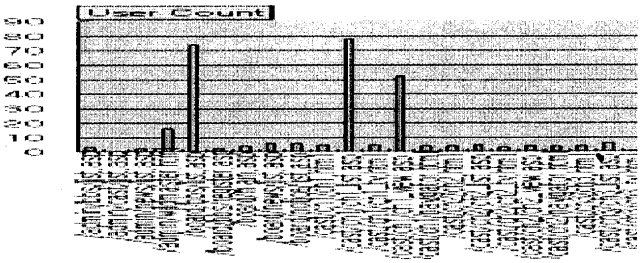

Figure 4. Page dimension (partial pages)

Index, registration, login and informative pages are popular as in Figure 4. But others' access rate is very low.

The time dimension of the second site is shown in Figure 5. Here, the visiting count of most attributes is a little dense. But if a site is sensitive to time (for example, ski resort sites or bathing place sites), sparse and dense attributes will definitely come out. Finally, Figure 6 shows sparse and dense attributes of the search engine dimension in the third site. Providing visitors with proper or a variety of search keywords makes the following result.

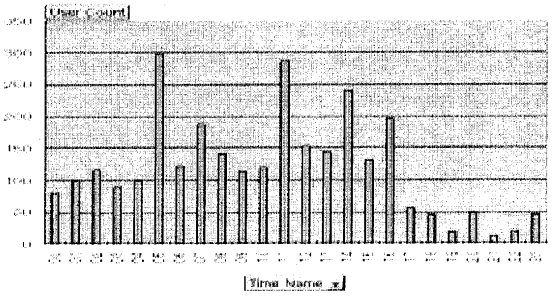

Figure 5. Time dimension

3.2 The sparsity pattern of web log data

3.2.1 Sparsity patterns in the two and the three dimensions.

To find sparsity patterns, we combined each dimension and visualized by 3D Cube Explore in DBMiner.

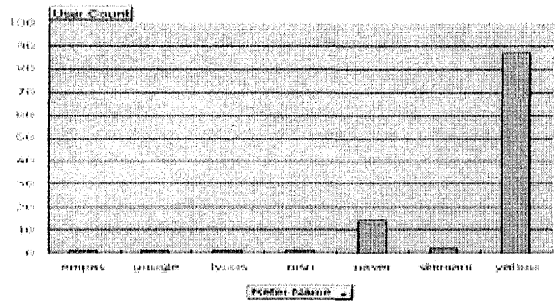

Figure 6. Search engine dimension

Figure 7 shows a sparsity pattern in the page and time dimension.

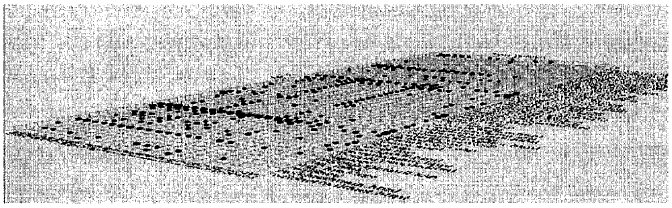

Figure 7. Sparsity pattern: the two dimensions

As in [9], if all dimensions in multidimensional data model have sparse relationship between them, we can find the following different sparsity patterns – random (A), stripe (B), cluster (C), slice (D).

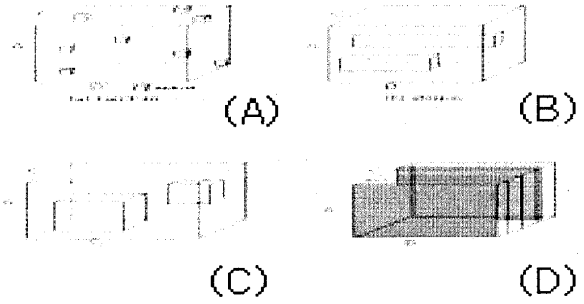

Figure 8. Sparsity patterns

In Figure 7, we can find a grid pattern that stripe patterns in Figure 8 are crossed. This occurs when informative and interesting pages are accessed at any time and also access rate of most pages is high at busy time. But if the order of dimension attributes is rearranged, cluster patterns can be also made. Figure 9 shows a sparsity pattern in the three dimensions – the page, search engine and time dimension. As the two dimensions, we can see gird and cluster patterns.

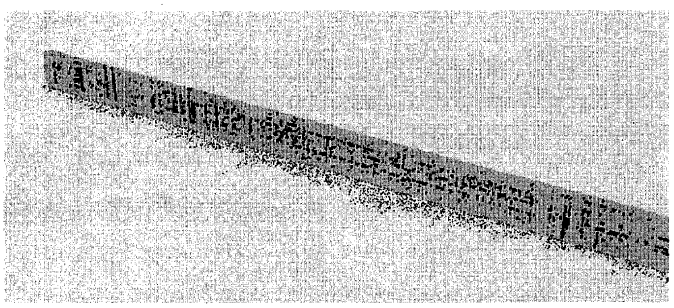

Figure 9. Sparsity pattern: the three dimensions

4. Data model and query model design for benchmark

In this section, we explain the data model and the query model for benchmark with sparsity patterns (grid and cluster).

4.1 The test data model

Dimensions are based on the visitor, the page and the time dimension. And a measure is a visiting count. Hierarchies and detail information of each dimension are shown in Table 4

Table 4. Two Dimension information

Dimension	Visitor	Page	Time
The number of Attributes	10,000	50	2,150
Hierarchy	1	4	3

With these dimensions and the measure, we built data models of the two and the three dimensions. Figure 10 and Table 5 show detail information for data model of the two dimensions.

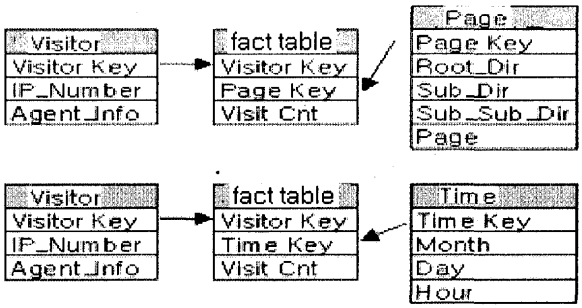

Figure 10. Data model: the two dimensions

The three dimensions are shown in Figure 11 and Table 5.

Table 5. Detail information: The Two / Three dimensions

Table 5. Three Dimension Information

Example	Dimensions	Total records	Sparsity(10%)
A	Visitor_Page	500,000	50,000
B	Visitor_Time	21,600,000	2,160,000
C	Visitor_Page_Time	1,080,000,000	108,000,000

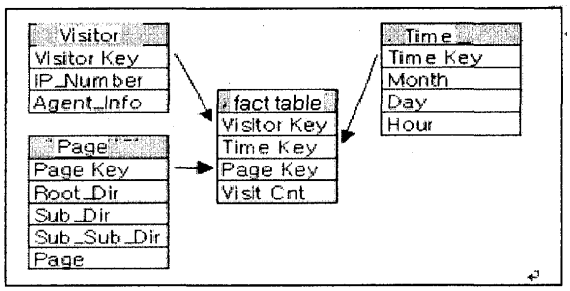

Figure 11. Data model: the three dimensions

4.2 Query model

In general, main objectives of queries in web log data are to understand the needs and preference of customers in order to win new visitors and retain existing visitors. So we set up the query

model as seven basic OLAP operations and three operations used frequently in web log data analysis. Basic OLAP operations of each data model are shown in Table 6,7,8.

Table 6. Query model: Example A

Exact Match	How often does a visitor (VK03018) visit an event page (PK00003)?
Slice	How often does a visitor (VK03018) visit every page?
Dice	How often does every visitor visit a menu (S00001)?
Range	How often do visitors (VK03000 ~ VK03600) visit a menu (S00001)?
Pivot	Show visitors on columns and pages on rows instead of pages on columns and visitors on rows
Roll Up	Show the visiting count grouped by the up level of a page (PK00003) in a visitor (VK03018)
Drill Down	Show the visiting count grouped by the down level of a directory (R00001) in a visitor (VK03018)

Queries used frequently web log data analysis are shown in Table 9

Table 7. Query model: Example B

Exact Match	How often does a visitor (VK03018) visit at TK00042?
Slice	How often does a visitor (VK03018) visit at every time?
Dice	How often does every visitor visit on 02/08?
Range	How often do visitors (VK03000 ~ VK03600) visit on 02/08?
Pivot	Show visitors on columns and time on rows instead of time on columns and visitors on rows
Roll Up	Show the visiting count grouped by the up level of day (02/08) in a visitor (VK03018)
Drill Down	Show the visiting count grouped by the down level of day (02/08) in a visitor (VK03018)

Table 8. Query model: Example C

Exact Match	How often does a visitor (VK00012) visit an event page (PK00003) at TK02041?
Slice	How often does a visitor (VK00001) visit every page at every time?
Dice	How often do visitors (VK00300 ~ VK00330) visit every page on 01/08
Range	How often do visitors (VK00300 ~ VK03300) visit pages (PK00001 ~ PK00030) on 01/08?
Pivot	Show pages on columns and time on rows instead of time on columns and pages on rows in a visitor (VK00012)
Roll Up	How often does a visitor (VK00012) visit a page (PK00008) on the up level (August) of the day (01/08)
Drill Down	How often does a visitor (VK00012) visit a page (PK00008) at the down level of the day (01/08)

Table 9. Query model: Top500 query

Visitor_Page	Show top 500 visitors that visit most frequently an event page (PK00003)
Visotor_Time	Show top 500 visitors that visit most frequently on October
Visitor_Page_Time	Show top 500 visitors that visit most frequently an event page (PK00003) on October

5. Benchmark system implementation

5.1 MOLAP system implementation using the chunking method

MOLAP systems store their data as sparse arrays and compute all of aggregates either in response to a user query, or as part of a "load process" that precomputes aggregates to speed later queries. But the array itself is far too large to fit in memory so that must be split up into *chunks*. We defined a sparse chunk in which less than 40 % of the array cells have a valid value and compressed using the *offset-value pair*. And computing aggregates is based on *A Single Pass Multi-way Array Cubing Algorithm* in [7]. This tries to reduce memory requirements by keeping only parts of the group-by arrays in memory and compute multiple group-bys simultaneously.

5.2 Benchmark system implementation

In this section, we will explain the system architecture for evaluating the performance of three OLAP systems. The system architecture is shown in Figure 12. As shown in Figure12, test data are imported in MS SQL 2000 analysis service and Oracle Express. And then each system creates and processes manually cubes using its cube wizard. In MS SQL 2000 analysis service, the benchmark program accesses the cube through ADO MD (ActiveX Data Object

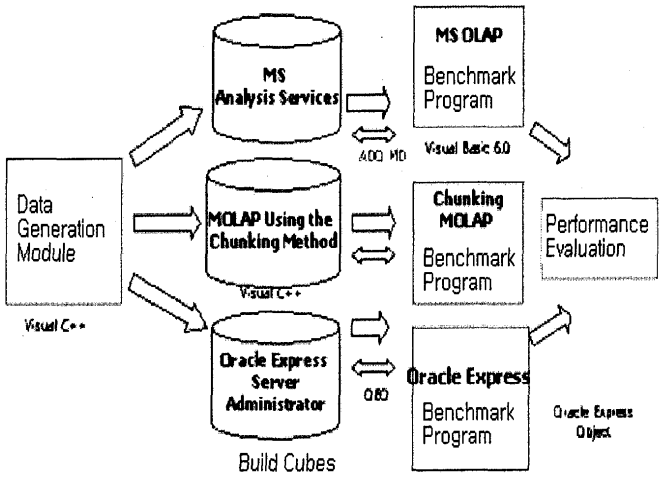

Figure 12. System architecture

Multi-dimensional) API and processes queries by MDX (Multi-dimensional Expression). That of Oracle Express 6.3.2 is developed by OEO (Oracle Express Object) and Oracle Basic

Language [9]. And the performance of these systems is evaluated by query execution time.

6. Performance results

At this time, we can show performance results and evaluate which system is more efficient for web log data analysis.

The performance of "Top 500 query" is affected by sorting algorithms. So only MS and Oracle except MOLAP with the chunking method are evaluated. In Figure 13, we can see that regardless of sparsity patterns, Oracle Express is a little better than MS SQL 2000 Analysis Service in Example A and Example B.

But In Example C, we can see that MS SQL 2000 Analysis Service is ten times better than Oracle Express Server.

Figure 14, 15, 16 shows that MS SQL 2000 Analysis Service is a little worse in dice and range operation that define a sub-cube by performing selections on two dimensions. But in dice and range operations for three dimensions and a pivot operation, Oracle Express Server shows the worst performance.

In conclusion, when you need to perform the pivot or dice or range operation or "Top 500 query" according to increase the dimension number, Oracle Express Server is much worse than the others.

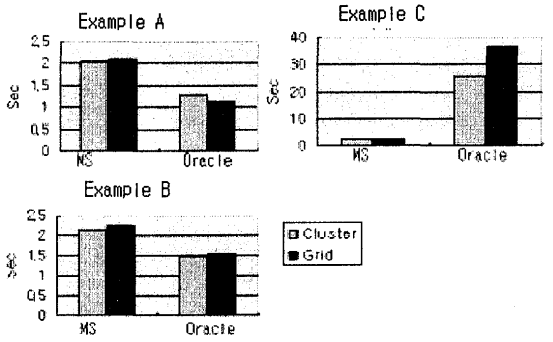

Figure 13. Performance results: Top 500 query

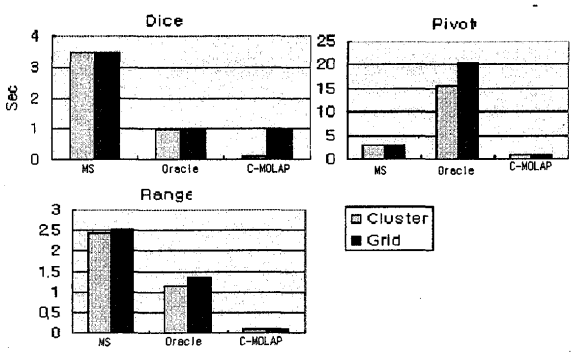

Figure 14. Performance results: Example A

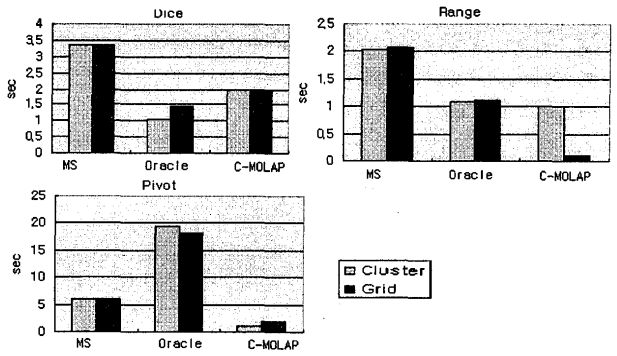

Figure 15. Performance results: Example B

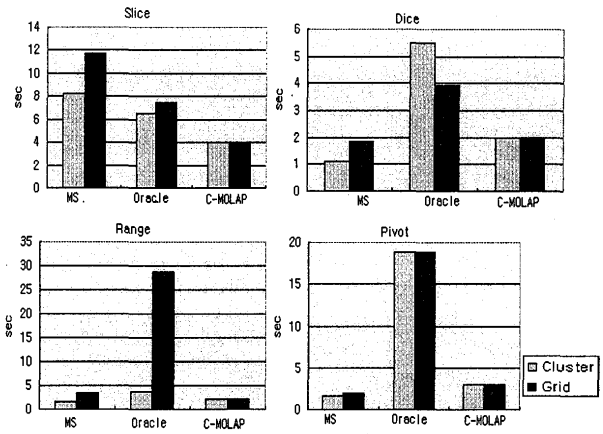

Figure 16. Performance results: Example C

7. Conclusions

Using preprocessed log data of three sites, we have found out why the sparsity occurs and what kinds of sparsity patterns in the two and the three dimensions. After test data was generated using the data model, seven OLAP operations and three "Top 500" queries were used to evaluate three OLAP systems in each sparsity pattern. Oracle Express shows a little better result than the others in dice, range and "Top 500 query" for two dimensions. But Oracle Express Server shows much worst performance when the number of dimension increases. We developed multidimensional storage system called C-MOLAP to handle sparse data and applied it to web log data and found its performance is competitive to commercial packages. As a further research, more improvement to design specialized algorithms to handle specific sparisty patterns is required.

8. References

[1] MaxScan Corp, White Paper. *An Accelerator for Click Stream Data Analysis Applications*

[2] Pilot Software, White Paper: *An Introduction to OLAP: Multidimensional Terminology and Technology*

[3] Sanjay Goil and Alok Choudhary: *Sparse Data Storage of Multi-Dimensional Data for OLAP and Data Mining*, Technical Report CPDC-TR-9801-005, Center for Parallel and Distributed Computing, Northwestern University, 1997

[4] White paper :
http://www.olapreport.com/DatabaseExplosion.htm
[5] Oracle Corp, *Sparsity Management System for Multidimensional Databases*, U.S. patent #5943677, Aug, 1999
[6] Robert J.Earle. Arbor Software Corporation, *Method and Apparatus for Storing and Retrieving Multidimensional Data in Computer Memory*, U.S.Patent #5359724, Oct. 1994
[7] Y.Zhao, P.M. Deshpande, and J.F.Naughton, *An Array-Based Algorithm for Simultaneous Multidimensional Aggregates*, In Proc.of ACM SIGMOD, pp 159-170, 1997
[8] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava: *Data Preparation for mining world wide web browsing patterns*, the Journal of Knowledge and Information System, Vol. 1, No. 1, 1999
[9] Ju-young Kang: *Classification of sparsity patterns and performance evaluation in OLAP system*, the Master's Thesis of Department of Computer Science and Engineering, Ewha Institute of Science and Technology, 2000
[10] Oracle Express Objects User's Guide Release 6.3, 1999