D30

$\mathbf{i^3}$: Intelligent, Interactive Investigaton of OLAP data cubes[*]

Sunita Sarawagi
School of Information Technology
Indian Institute of Technology, Bombay
Mumbai 400076, INDIA
Phone: +91-22-576-7904, Fax: +91-22-576-7902
sunita@cs.berkeley.edu

GOAL

The goal of the $\mathbf{i^3}$(eye cube) project is to enhance multidimensional database products with a suite of advanced operators to automate data analysis tasks that are currently handled through manual exploration. Most OLAP products are rather simplistic and rely heavily on the user's intuition to manually drive the discovery process. Such ad hoc user-driven exploration gets tedious and error-prone as data dimensionality and size increases. We propose to automate them using advanced operators that can be invoked interactively like existing simple operators yet significantly reduce the reliance on manual discovery.

Introduction

Online Analytical Processing (OLAP) products were developed to help analysts do decision support on historic transactional data. Logically, they expose a multidimensional view of the data with categorical attributes like Products and Geographies forming the *dimensions* and numeric attributes like Sales and Revenue forming the *measures* or cells of the multidimensional cube. Dimensions usually have associated with them *hierarchies* that specify aggregation levels. The measure attributes are aggregated to various levels of detail of the combination of dimension attributes using functions like sum, average, max, min, count and variance.

For exploring the multidimensional data cube there are navigational operators like select, drill-down, roll-up and pivot conforming to the multidimensional view of data. These have fast response times so that the user can interactively invoke sequences of such operations. A typical OLAP analysis sequence proceeds as follows: the user starts from an aggregated level, inspects the entries visually maybe aided by some graphical tools, selects subsets to inspect further based on some intuitive hypothesis or needs, drills down to more detail, inspects the entries again and either rolls up to some less detailed view or drills down further and so on.

The above form of manual exploration can get tedious and error-prone for large datasets that commonly appear in real-life. For instance, a typical OLAP dataset may have five to seven dimensions, between two and seven levels hierarchy on each dimension and aggregates more than a million rows. The analyst could potentially overlook significant discoveries due to the heavy reliance on intuition and visual inspection.

The goal of the $\mathbf{i^3}$ ("eye-cube") is to take these OLAP tools to the next stage of interactive analysis where we automate much of the manual effort spent in analysis. Mining technology can play a fitting role in improving the state of these products. The huge size of the data and the rich star-like structure based on dimensions and hierarchies make it possible to integrate mining tools in useful and innovative ways. Existing tools like classification, clustering, association rules and time series analysis are starting to be deployed,

[*]Submitted for publication. Copyright and intellectual property rights currently with IIT Bombay

Product	Platform	Geography	Year
Product name (67)	Platform name (43)	Geography (4)	Year (5)
Prod_Category (14)	Plat_Type (6)		
Prod_Group (3)	Plat_User (2)		

Figure 1: Dimensions and hierarchies of the software revenue data. The number in brackets indicate the size of that level of the dimension.

Platform	(All)				
Product	(All)				
Sum of Revenue	Year				
Geography	1990	1991	1992	1993	1994
Asia/Pacific	1440.24	1946.82	3453.56	5576.35	6309.88
Rest of World	2170.02	2154.14	4577.42	5203.84	5510.09
United States	6545.49	7524.29	10946.87	13545.42	15817.18
Western Europe	4551.90	6061.23	10053.19	12577.50	13501.03

Figure 2: Total revenue by geography and year. The two boxes with dark boundaries ('Rest of World', year 1990 and 1991) indicate the two values being compared.

albeit sparingly. But the approach so far has been to take existing mining algorithms and integrate them within OLAP products. Our approach is different in that we first investigate how and why analysts currently explore the data cube and next automate them using new or previously known operators. Unlike the batch processing of existing mining algorithms we wish to enable interactive invocation so that an analyst can use them seamlessly with the existing simple operations. Interactive exploration has the advantage that the analyst's external-world insight can be combined with the discoveries obtained purely from data.

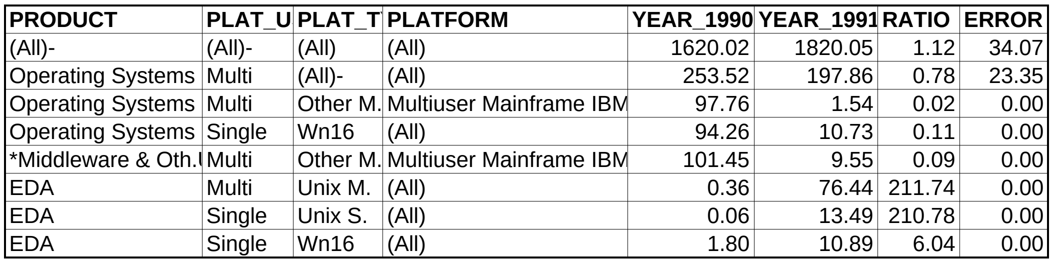
Project description

Our proposed suite of extensions appear in the form of a toolkit attached with a OLAP product. At this demo we will present three such operators: DIFF, and SURPRISE. For ease of understanding, we will illustrate the working of the our prototype using a real-life dataset obtained from International Data Corporation (IDC) about the total yearly revenue in millions of dollars for different software products from 1990 to 1994. The schema as shown in Figure 1 consists of four dimensions Product, Platform, Geography and Time and a three level hierarchy on the Product and Platform dimension.

The DIFF operator

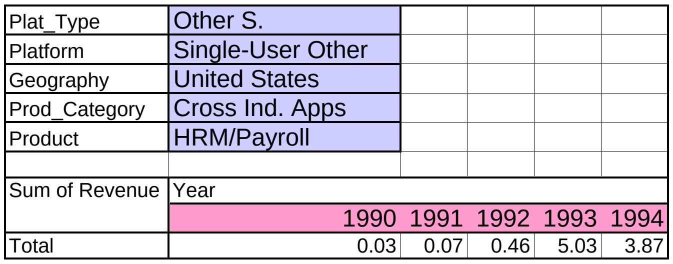
One reason why analysts manually drill down to explore detailed data is to find causes for drops or increases observed at an aggregated level. For automating this task we propose the DIFF operator that can report a summarized difference between two values observed at aggregate levels. For example, a busy executive looking at the annual revenues across different geographies (Figure 2) might quickly wish to find the main reasons why sales dropped from 1990 to 1991 for the region 'Rest of the World'. By drilling onto details of individual sales figures perhaps he can pin point those values which contributed to the drop. But this could be daunting when the number of values contributing to the top-level aggregate is large. Our proposed DIFF operator can be used in such cases. In a single step it will do all the digging for him and return the main reasons in a compact form that he can easily assimilate.

We use a information theoretic formulation for summarizing the detailed differences to trade off the size of the answer with the loss in accuracy due to summarization of multiple similar changes. In Figure 3 we show a sample answer returned by our operator in response to the drop observed in Figure 2. In the figure the first row shows that after discounting the rows explicitly mentioned below it as indicated by the "-" symbol after (All), the overall revenue actually increased by 10%. The next four rows (rows second through fifth) identify entries that were largely responsible for the big drop. For instance, for "Operating systems" on "Multiuser



PRODUCT	PLAT_U	PLAT_T	PLATFORM	YEAR_1990	YEAR_1991	RATIO	ERROR
(All)-	(All)-	(All)	(All)	1620.02	1820.05	1.12	34.07
Operating Systems	Multi	(All)-	(All)	253.52	197.86	0.78	23.35
Operating Systems	Multi	Other M.	Multiuser Mainframe IBM	97.76	1.54	0.02	0.00
Operating Systems	Single	Wn16	(All)	94.26	10.73	0.11	0.00
*Middleware & Oth.U	Multi	Other M.	Multiuser Mainframe IBM	101.45	9.55	0.09	0.00
EDA	Multi	Unix M.	(All)	0.36	76.44	211.74	0.00
EDA	Single	Unix S.	(All)	0.06	13.49	210.78	0.00
EDA	Single	Wn16	(All)	1.80	10.89	6.04	0.00

Figure 3: Reasons for the drop in revenue marked in Figure 2.



Plat_Type	Other S.				
Platform	Single-User Other				
Geography	United States				
Prod_Category	Cross Ind. Apps				
Product	HRM/Payroll				
Sum of Revenue	Year				
	1990	1991	1992	1993	1994
Total	0.03	0.07	0.46	5.03	3.87

Figure 4: A problematic drop in revenue from 1993 to 1994 observed for Product = “HRM/Payroll”, Geography = “United States” and Platform = “Single-User Other”.

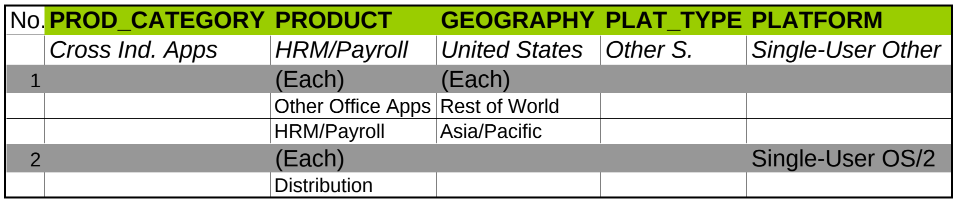
Mainframe IBM” the revenue decreased from 97 to 1.5, a factor of 60 reduction and for “Operating systems” and all platforms of type “Wn16” the total revenue dropped from 94.3 to 10.7. The last three rows show cases where the revenue increased by significantly more than the 10% overall increase indicated by the first row.

More details of this work appeared in the 25th International Conference for Very Large Databases, 1999.

The RELAX operator

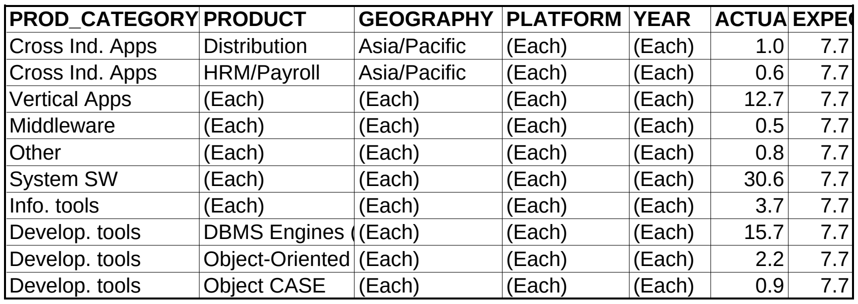
Suppose a local branch manager exploring his subportion of the data cube notices a significant drop in sales somewhere in detailed data. Often, the next step taken by him is to investigate whether this was the only case where such a drop was observed or is this change part of a bigger problem affecting other cases too. For this he rolls up to the next level and views the problem case in context of combinations of other dimensions using a succession of selection, drill-down and pivot steps. The RELAX operator can be used to automate this search. The output of the operator is a set of maximal regions around the problem case where a similar change was observed.

For instance in Figure 4 an analyst is viewing the total revenues along time for a fixed Geography, Platform and Product when he observes a problematic drop from 1993 to 1994. He might next view this case in context of other Geographies, Platforms or Products to check if one or more of them had similar drop and explore further out around some of these identified problem cases trying to find a pattern. We automate this search through the RELAX operator. Figure 5 shows the output of the operator around the problem case of Figure 4. The first relaxation shown by the grayed row numbered “1” informs that we can generalize simultaneously along the Geography and Product dimension for the same platform and Prod_Category. That is, *each Product* in Category “Cross-Ind. Apps” in *each Geography* had a drop from 1993 to 1994. The two exceptions to this generalization are listed immediately below. The second relaxation shown by the grayed row numbered “2”, further away from the problem case than the first one, states that within the same Platform_type (“Other S.”), another platform “Single-User OS/2” had a similar drop for each Product (except Product “Distribution”) for the same Category and Geography. Armed with this exact extent and scope of the problem, the analyst can then apply his insight to infer on possible external reasons.



No.	PROD_CATEGORY	PRODUCT	GEOGRAPHY	PLAT_TYPE	PLATFORM
	Cross Ind. Apps	*HRM/Payroll*	*United States*	*Other S.*	*Single-User Other*
1		(Each)	(Each)		
		Other Office Apps	Rest of World		
		HRM/Payroll	Asia/Pacific		
2		(Each)			Single-User OS/2
		Distribution			

Figure 5: Relaxations from the problem case in Figure 4.



PROD_CATEGORY	PRODUCT	GEOGRAPHY	PLATFORM	YEAR	ACTUA	EXPE
Cross Ind. Apps	Distribution	Asia/Pacific	(Each)	(Each)	1.0	7.7
Cross Ind. Apps	HRM/Payroll	Asia/Pacific	(Each)	(Each)	0.6	7.7
Vertical Apps	(Each)	(Each)	(Each)	(Each)	12.7	7.7
Middleware	(Each)	(Each)	(Each)	(Each)	0.5	7.7
Other	(Each)	(Each)	(Each)	(Each)	0.8	7.7
System SW	(Each)	(Each)	(Each)	(Each)	30.6	7.7
Info. tools	(Each)	(Each)	(Each)	(Each)	3.7	7.7
Develop. tools	DBMS Engines (	(Each)	(Each)	(Each)	15.7	7.7
Develop. tools	Object-Oriented	(Each)	(Each)	(Each)	2.2	7.7
Develop. tools	Object CASE	(Each)	(Each)	(Each)	0.9	7.7

Figure 6: Informative regions returned to user who has not viewed any part of the data.

The SURPRISE operator

This operator is geared towards helping a user quickly familiarize himself with the significant features of a OLAP data cube in a manner that adapts to the prior context of the user. We developed a user personalization tool consisting of the following three pieces. First, we capture the analysts context by continuously keeping track of the parts of the cube that he is familiar with using either transparent logs of various facets of the cube he visited or explicit entries by the user. Next, we use an asynchronous process to connect together the various visited pieces of the cube to model the user's mental expectation of the rest of the cube. Finally, at any time, the user can invoke the SURPRISE operator that will output the most interesting unvisited parts of the cube given his established context. This process of updating the user's context based on visited parts and querying for regions to explore further continues in a loop until the user's mental model perfectly matches the actual cube.

We show two sample outputs of the SURPRISE operator — one for the case of a user who is totally unfamiliar with the dataset, second for a user who is totally familiar with old data from year 1990 to 1992, but for the last two years (1993 and 1994) has only seen the total revenue not the detail break up. The first user assumes that the revenue for each case is the same not knowing anything else about the data. The output to him (as shown in Figure 6) is a list of Product, Geography combination where that assumption is most violated. In the figure, the last two columns denote the actual and the user expected values respectively. The seond user based on the totals along year expects to see an increase in revenue from 1992 to 1993 to 1994 for each Product,Platform,Geography combination. The output to him as shown in Figure 7 is a set of individual cases that dropped or increased significantly in 1993 or 1994. Thus, the second user will only have to concentrate on those cases which cannot be extrapolated based on his experience of past data and the yearly totals.

The key pieces of technology implemented for this paradigm are (1) a method for finding expected values of unvisited parts from visited parts of the cube and (2) a method for defining what is interesting to explore further. We formulate the first problem using the maximum entropy principle. Accordingly, we report as interesting those regions which if visited by the user will lead to the smallest gap between the new expected values and the actual values. We have developed procedures for *incrementally* and efficiently updating the interesting frontiers to explore further as new areas of the cube are visited.

PRODUCT	GEOGRAPHY	PLATFORM	YEAR	ACTUAl	EXPEC
Other Office Apps	Western Europe	Multiuser Mainframe IBM	1993	3.05	78.25
EDA	Western Europe	Multiuser UNIX	1994	19.97	142.41
Operating Systems	United States	Multiuser Other Server	1994	0.22	47.82
Operating Systems	Western Europe	Multiuser Minicomputer O	1993	3.27	69.74
Middleware	Asia/Pacific	Multiuser Mainframe IBM	1993	185.72	96.10
CASE (Non-Object)	United States	16-bit Windows/DOS	1994	1.77	57.58
DBMS Engines (Non-Object)	United States	Multiuser Mainframe IBM	1994	90.31	315.03
DBMS Engines (Non-Object)	Rest of World	Multiuser Mainframe IBM	1994	11.00	98.92
4GL & Report Writers	Rest of World	Multiuser Mainframe IBM	1993	0.31	42.48
3GLs & Develop. Environments	Rest of World	Multiuser Mainframe IBM	1993	0.62	38.67

Figure 7: Informative regions returned to user who is familiar with the entire cube for years 1990-1992 but unfamiliar with years 1993 and 1994.

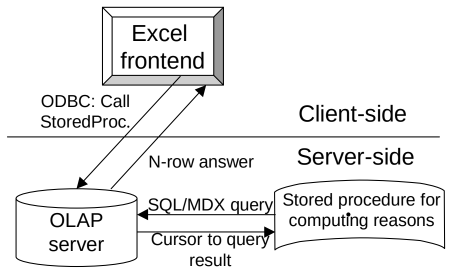


Figure 8: Our prototype.

Integrating with OLAP products

Our operators are designed to fit as add ons to existing OLAP systems. Our current prototype is based on IBM's DB2/UDB database that we view as a ROLAP engine with Microsoft Excel's Pivot Table as the OLAP frontend. The algorithms for each of these operators are packaged as java stored procedures that reside on the server side. The client uses ODBC embedded in Visual Basic for invoking the stored procedure. The answers are stored in the database and accessed by the client using ODBC. In Figure 8 we show the architecture for our prototype. Each of the three operators use dynamically generated queries to push all heavy weight processing to the DBMS and thus ride on the OLAP servers indexing and query processing capabilities. Also, our implementation does not not rely on any off line index construction or preprocessing steps. We can directly work with any available OLAP data source. The data is assumed to be laid out in a star schema for the processing. The operators are implemented for interactive invocation and we have validated this using experiments on the industry OLAP benchmark (with 1.36 million tuples). For instance, the DIFF operator can process more than a quarter million records in less than 10 seconds and scales gracefully with increasing dimensions and number of attributes. We also plan to test our operators as attachments to Oracle Express and Microsoft SQL Server's OLAP extensions.

Conclusion

Multidimensional OLAP products provide an excellent opportunity for integrating mining functionality because of their widespread acceptance as a decision support tool and their existing heavy reliance on manual, user-driven analysis. In this project we demonstrated with the help of three operators how this can be done. In each case, we automate some existing tedious, manual discovery process by introducing a new operator.

We have fast algorithms and efficient implementations of the operators for *interactive* use during cube exploration. We have also integrated them with existing OLAP products and can work as add ons thus allowing easy deployment.

Future scope of the project includes identifying newer operators, exploring use of these operators in normal relational databases and investigating ways of inter-operating between them. Also we view this project as a precursor to interactive, adhoc mining fully implemented inside a database system.