Integrated Performance and Visualization Enhancements of OLAP Using Growing Self Organizing Neural Networks

Muhammad Usman
Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad, Pakistan
Email: usmanspak@yahoo.com

Sohail Asghar
Mohammad Ali Jinnah University, Islamabad, Pakistan
Email: sohail.asghar@jinnah.edu.pk

Simon Fong
University of Macau, Taipa, Macau SAR
Email: ccfong@umac.mo

***Abstract*—OLAP performance and its data visualization can be improved using different types of enhancement techniques. Previous research has taken two separate directions in OLAP performance improvement and visualization enhancement respectively. Some recent works have shown the benefits of combining OLAP and Data Mining. Our previous work presents an architecture for the enhancement of OLAP functionality by integrating OLAP and Data Mining. In this paper, we proposed a novel architecture that not only overcomes the existing limitations, but also provides a way for an integrated enhancement of performance and visualization using self organizing neural network. We have developed a prototype and validated the proposed architecture using real-life data sets. Experimental results show that cube construction time and its interactive data visualization capability can be improved remarkably. By integrating enhanced OLAP with data mining system a higher degree of enhancement is achieved which makes significant advancement in the modern OLAP systems.**

***Index Terms*— Clustering, Data Mining, GSOM, Multi-dimensional Data, OLAP, Performance Enhancement, Visualization Techniques**

I. Introduction

OLAP technology refers to a set of data analysis techniques to view the data from all of the transactional systems in an interactive way in order to support the decision-making process. The fast growing complexity and volumes of the data to be analyzed impose new requirements on OLAP systems [1]. An OLAP system's performance and level of data visualization can be enhanced using different tools and techniques. With the coupling of these enhancement techniques, OLAP functionality can be enhanced [2]. However, OLAP performance improvement and visualization enhancement have been taken separately in the past.

Figure 1 depicts the integration of performance improvement and visualization enhancement practices. Another aspect of enhancement is Data Mining, which aims at the extraction of synthesized and previously unknown insights from large databases [3]. It can be viewed as an automated application of algorithms to detect patterns and extract knowledge from data that is not obvious to the user [4]. Some recent work has shown the benefits of combining OLAP and Data Mining. According to [5], automated techniques of Data Mining can make OLAP more useful and easier to apply in the overall scheme of decision support systems. Furthermore, Data mining techniques like Associations [6], Classification [7], Clustering [8] and Trend Analysis [9] can be used together with OLAP to discover knowledge from data [10].

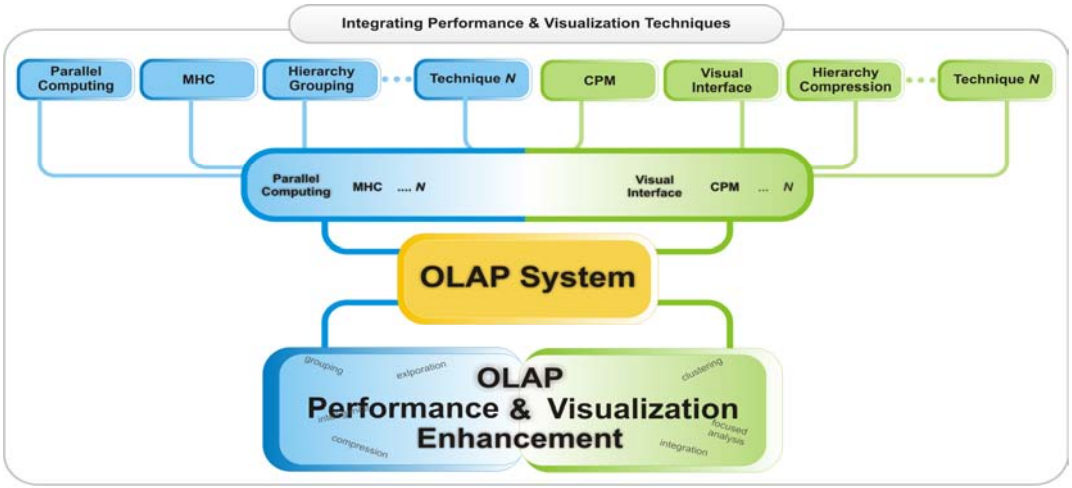


Figure 1. Integration of Enhancement Techniques [2]

Manuscript received August 11, 2009; revised October 1, 2009.

doi:10.4304/jait.1.1.26-37

In the quest of OLAP enhancement research, Asghar [3] proposed a functionality-enhancement technique using self-organizing neural networks. This technique proposed the integration of Data Mining with OLAP by passing the mined data to the OLAP engine for a more focused analysis and, hence, added intelligence to the OLAP system. The major limitation of the proposed enhancement architecture was the deficiency of work in the enhancement of OLAP performance and visualization. No visualization enhancement technique was used for the expanded view of the OLAP data. Data cube processing time and its physical drive storage were also not discussed. Users of the proposed architecture had to formulate queries to manually retrieve the data of their choice. Interactive visual analysis was missing which is a very attractive functionality of OLAP systems. Typical OLAP data do not change, as they are usually historical data. A major concern is often the support of ad-hoc data exploration by an analyst or other users looking for trends or patterns at v levels of details, perhaps integrated with decision support applications [10].

In this paper, we extend the previous work to overcome the existing limitations and provide an enhanced architecture that can cater for both performance improvement and visualization enhancement. The newly proposed architecture has various modules which allow the integrated performance and visualization enhancement of the OLAP system.

We developed an OLAP prototype system using C# language and other software development tools such as Microsoft SQL server [11], Microsoft Analysis Services [12] and Dundas OLAP services for Windows Form [13]. Experiments are done with data from Forest Cover Type [14] and Zoo [15] data set. It is observed that using the proposed architecture we can enhance the existing OLAP systems in terms of performance and visualization and can get a higher degree of overall enhancement by integrating the performance improvement and visualization enhancement techniques. To the best of our knowledge, no such architecture which features an enhancement solution for OLAP systems for improving performance and enhancing visualization capabilities was ever proposed. Our experimental results show that the cube construction time can be improved remarkably by using the clustered data tables as compared to relational tables. Similarly, by implementing various visualization and enhancement tools and APIs [16] at the OLAP systems can improve the level of interactive data visualization through the use of different types of charts, graphs and data grids.

The remaining of this paper is organized as follows: Section 2 highlights the summary of the past work on which our solution is based. Section 3 addresses the related work. We elaborate the proposed architecture of enhanced OLAP in section 4. The design is followed by description of implementation of our prototype in section 5. Section 6 is where experimental results are disused and compared with the previous work. A conclusions and possible future research directions are drawn at the end.

II. PREVIOUS WORK

This section provides a brief summary of our previous work about OLAP functionality enhancement, on which our proposed solution is built upon. In our previous work, we extended the capability of the OLAP systems by the use of a neural network. In addition to the usual visualization capabilities, it provided users with the opportunity to analyze data in clusters at different levels of abstraction. The technique used is basically called Growing Self-Organizing Map (GSOM) [17]. GSOM has been developed as a flexible data mining feature mapping method over the traditional Self-Organizing Map (SOM) [18]. The innovation of GSOM is the ability of generating feature maps of different levels of data abstraction using a parameter called 'spread factor'. This spread factor is initially used for generation of hierarchical clusters and analysis technique which is known as dynamic SOM Tree.

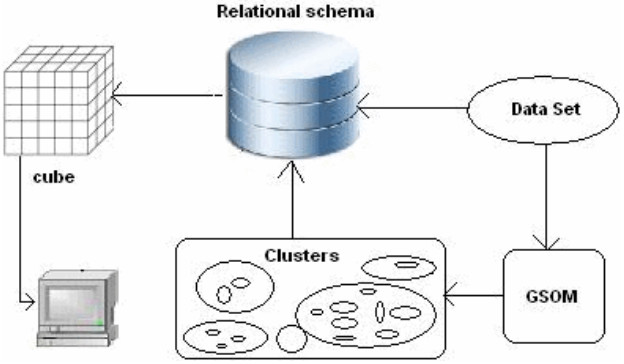


Figure 2. Architecture Proposed in the Previous Work [3].

These hierarchical clusters from the dynamic SOM Tree are subsequently used to provide the OLAP user with the ability to visualize and select data clusters at different levels of abstraction for further detailed analysis. Figure 2 depicts the previously proposed architecture for enhancing OLAP's functionality. This architecture indicates two different approaches to pass the data set to an OLAP engine. The framework we devised and presented in this paper is based on the hierarchical clusters generated by GSOM which are then translated into manual relational tables. The tables are stored in the relational database which serves the data source for the OLAP engine.

The architecture in the past work however has a number of drawbacks. Firstly, the clusters generated from GSOM are to be manually translated into relational tables. This means that user involvement is required for the clusters to be mapped on to a relational schema. Secondly, the OLAP user is unable to perform interactive visualization on the clustered data as there is no such facility available at the front-end. Customized queries are required to view the clustered data which requires knowledge of the complete clustered data in advance. It also lacks of support of Multidimensional expressions (MDX) which is a query language for the OLAP database. Users have to rely on the SQL command, 'GROUP BY' clause to perform runtime aggregation of

data. Cube construction time thus grows unfavorably as the increase of size of the data sets.

The motivation of this paper establishes from the need to keep up with the pace of incremental and fast OLAP development, and the limitations at the previous architecture specifically in terms of its performance and data visualization.

III. RELATED WORKS

As far as performance is concerned in OLAP system, the bottleneck usually involves data processing speeds over the structures of the data cubes. The authors in [5] identified the problem that OLAP operations require complex queries on underlying data, which can be very expensive in terms of computation time. Using parallel computers is one solution. Another approach which is similar to our solution proposed here is to improve cube processing time rather than the OLAP query processing time. For instance, authors in [19] suggested that OLAP performance can be improved by using the (MHC) Multi-dimensional Hierarchical Clustering technique. Clustering was introduced as a way to speed up aggregation queries without additional storage cost for view materialization.

In contrast, our work is to have used GSOM for generating hierarchical clusters for a focused analysis instead of speeding up OLAP queries. Similarly, authors in [20] achieved Heuristic Optimization of OLAP in MHC (multi-dimensionally hierarchically clustered) databases. They found that commercial relational database management systems use multiple one-dimensional indexes to process OLAP queries that restrict multiple dimensions. They presented architecture for MHC databases based on CSB star schema. In our work, we adopted this concept of using relational tables that contain clustered hierarchical information, that are transformed into typical star schema. Along with this concept, researchers in [21] suggested an enhanced OLAP operator based on the Agglomerative Hierarchical Clustering (AHC). The operator is called Operator for Aggregation by Clustering (OpAC) that is able to provide significant aggregates of facts referred to complex objects. Our approach is slightly different that we do not use any operator for clustering. Instead, we use a separate analysis server to construct a cube using the star schema source residing in the database server.

There are other innovated techniques for cube enhancements, such as Cube Presentation Model (CPM) recommended in [22]. CPM can be naturally mapped with an advanced visualization technique called Table Lens. A visual interface for exploring OLAP data with coordinated-dimension hierarchies is introduced in [10].

In literature, a lot of works were devoted to visualization enhancement techniques. Just to name a few, an advanced tool (CommonGIS) for highly interactive visual exploration of spatial data is in [23]. Followed by that, authors in [24] extended a tool for Spatial OLAP, called SOVAT (Spatial OLAP Visualization and Analysis Tool). A hierarchy-driven compression technique for the advanced visualization of multi-dimensional cubes is suggested in [25]. Then a new visual interactive exploration technique for OLAP is presented in [26]. This is similar to our work in terms of OLAP user facilitation. This enhanced architecture, allows novice users of OLAP technology to explore and analyze OLAP data cubes without sophisticated queries.

Subsequently a framework is proposed in [27] for querying complex multi-dimensional data and transforming irregular hierarchies to make them navigable in a uniform manner. Lately in 2008, Mansmann [28] introduced a comprehensive visual exploration framework which implements OLAP operation as a form of powerful data navigation and allows users to explore data using a variety of interactive visualization techniques. The *Dundas* visualization toolkit which we have used in our work to visualize the OLAP data also allows user to view the data using a number of charts. The use of this software makes our work similar as our choice of visualization of data also provides a number of visualization views to understand and analyze the data in an interactive way.

As observed from our review, research communities advocate that associating Data Mining to OLAP is useful for rich analysis and OLAP tools [21]. By following this fusion idea, we emphasize the coupling of performance and visualization enhancement of OLAP systems as our solution. Why then is there a need for OLAP enhancement architecture? Though a number of enhancement architectures were proposed in the past, none of the work so far was intended towards integrated enhancement of both OLAP performance and visualization, and hence a strong need exists for it [25].

IV. PROPOSED ARCHITECTURE FOR ENHANCED OLAP

To fulfill the growing demands of OLAP users [3], a standardized architecture are required which can easily be deployed as a complete system, it can support the integrated enhancement. Figure 3 depicts such architecture, which enforces integrated enhancement of OLAP's performance and visualization.

We describe the important components of the proposed architecture for the enhancement of OLAP systems. The goal of this architecture is to integrate OLAP with Data Mining and to provide integrated performance and visualization enhancement. To achieve this objective, we deployed separate servers; one is for the database and the other one for the OLAP data. This architecture indicates two channels to pass data to the OLAP engine. The first path is the conventional or non-clustered method where a data set is loaded directly into the database server through Extract, Transform and Load (ETL) process. The data are stored in the form of a relational database. From the relational database a star schema is designed using standard SQL queries and data is loaded into the star schema. The OLAP server takes this star schema as a source to construct OLAP cubes. It also provides storage and management mechanism for the cube data. At the front-end a visualization tool captures the cubes generated by the OLAP server and displays the data in form of charts, reports and tables.

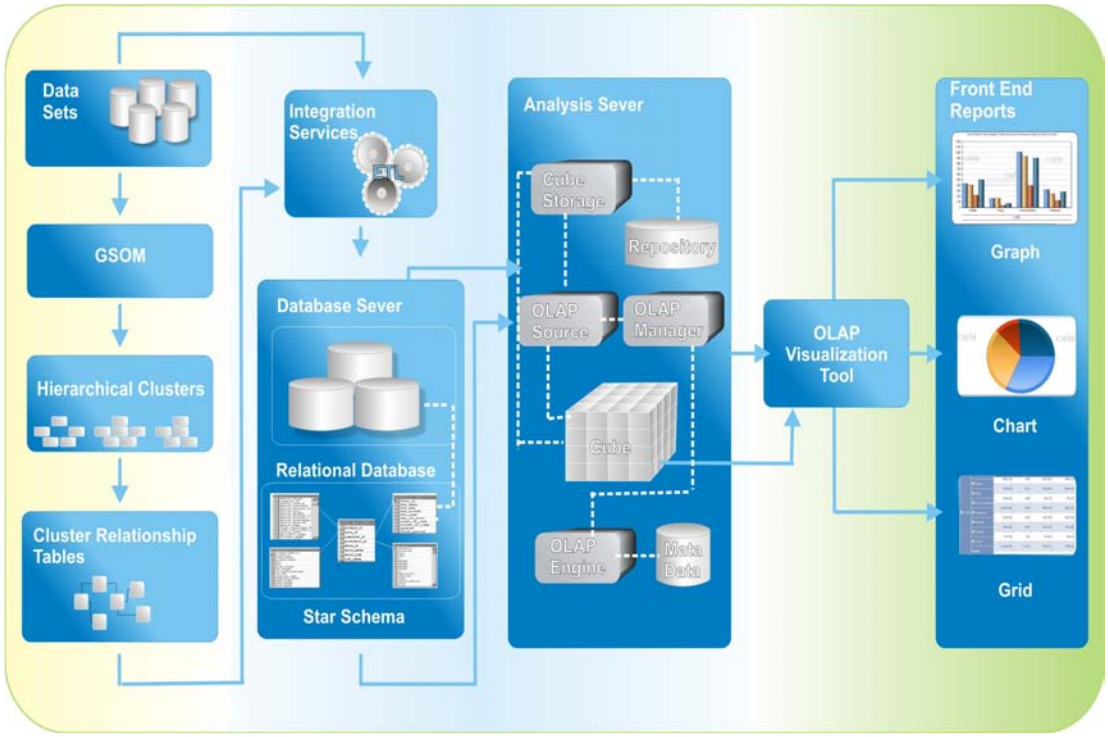


Figure 3. Enhanced OLAP Architecture Proposed.

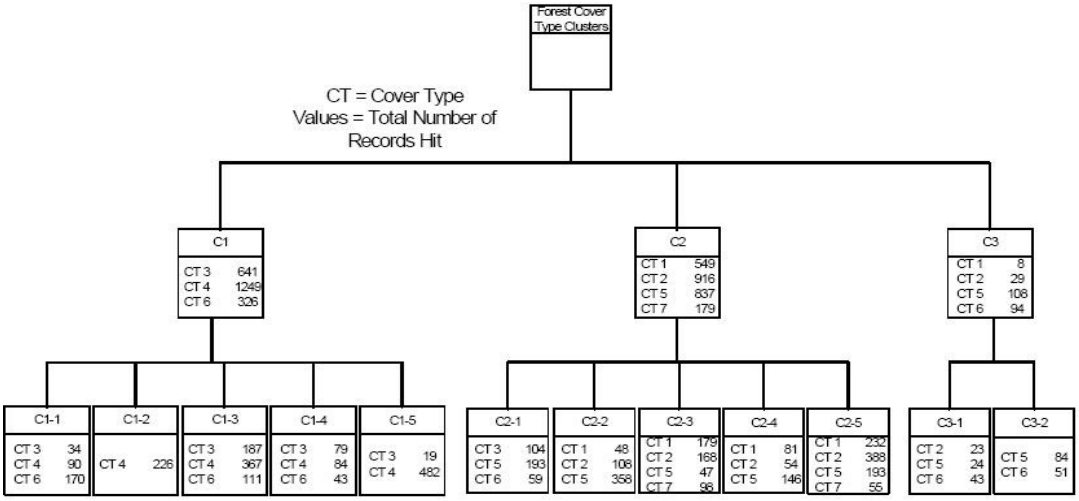


Figure 4. Hierarchical Cluster Decomposition of Forest data set.

A. *Data Processing*

One unique feature about the architecture we have devised to enhance OLAP is the hierarchical data clusters generated by GSOM. GSOM is based on the design of an unsupervised neural network [29]. The use of GSOM is mainly to produce the hierarchal clusters.

Data set is first fed to GSOM tool which produces the hierarchical clusters using a numerical spread factor. User can set the spread factor to control the number of hierarchical clusters generation using GSOM. An example is given in Figure 4 where the spread factor is 3. Once the clusters are generated the clusters are mapped manually into relational tables. The relational tables are stored in a database in the database server. From these relational tables star schema is created and uploaded with data. As mentioned previously the star schema becomes the data source. Using this source, cubes of the clustered data are constructed. These clustered cubes become the source of data to be visualized using the prototype in the same manner. From the visualized clusters, user can select the clusters of choice and perform analysis on particular clusters instead of the complete set of data.

B. *Visualization*

A Front-end visualization tool is connected with the OLAP server that generates cubes using star schema as a data OLAP cubes, displaying the cube data in various views such as charts, graphs, reports and tables at the user's end. Users can perform basic OLAP operations at will using front-end tool; hence, interactive analysis is available using drill down, roll up and other standard

OLAP operations. Visualization capabilities of both traditional and clustered OLAP data are enhanced. The proposed architecture achieves its objective as it integrates OLAP with Data Dining and uses clustered data for performance improvement in terms of cube processing time. In addition to this, the Visualization tool at the front end supports an interactive visual exploration of OLAP data using drill-down, roll-up charts and tables. The visualization tool enhances the visualization capabilities of both traditional and clustered OLAP data.

V. IMPLEMENTATION

This section is presented in two phases: in the first phase we explain the implementation details of the proposed architecture; in the second phase we describe the experiments performed on two different data sets.

Our architecture supports two data loading approaches, clustered and non-clustered. For the non-clustered approach which is the ordinary one, the data set stored in a form of single comma separated text file (depicted in Figure 5) is uploaded in a database as a table using Microsoft SQL Server 2000's Data Transformation Services (DTS) as shown in Figure 6.

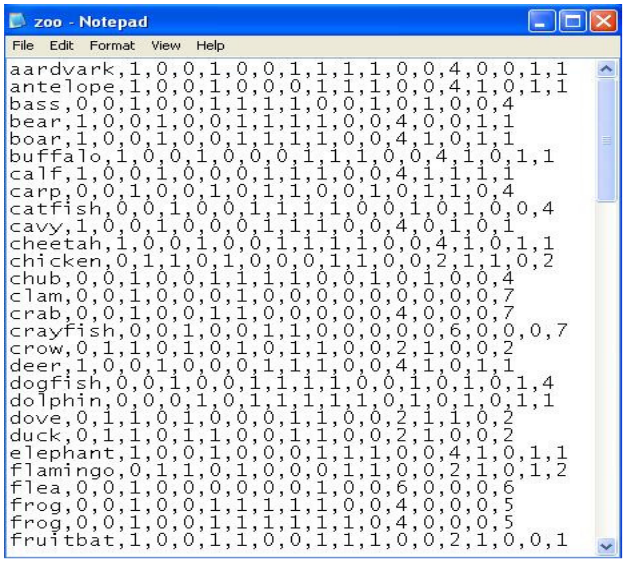


Figure 5. Data in text file.

After the database has been uploaded with the data, using SQL queries a single *fact table* and three *dimension* tables are created and uploaded with data to form a *star schema* as shown in Figure 7. This star schema residing in the database server is a source data for OLAP server. In this project we are using *Microsoft Analysis Services* as an OLAP data server. The OLAP engine in the Analysis server uses this data source to construct cubes.

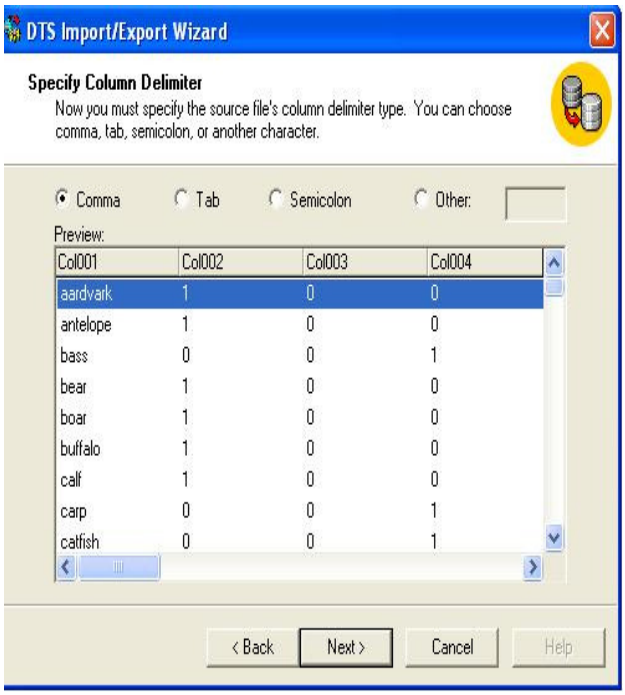


Figure 6. Data Transformation Services (DTS) Wizard View.

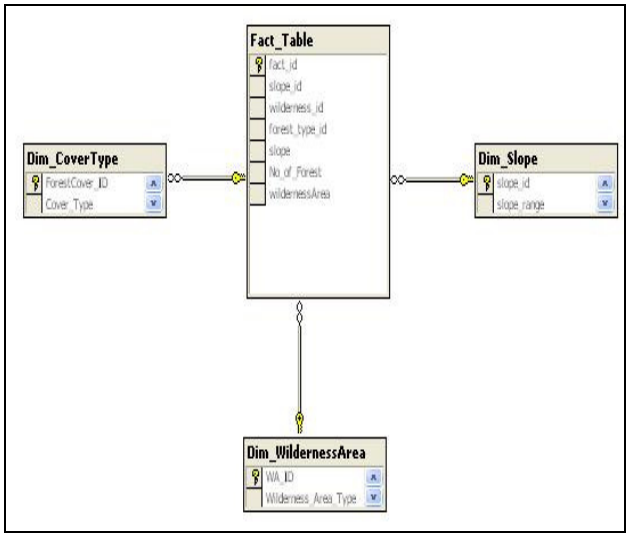


Figure 7. Star Schema of 'Forest Cover Type' Dataset.

Figure 8 shows the cube wizard of MS Analysis Services. Using this wizard user can construct cubes from the data by selecting the different *measures (facts)* available in the *fact table*.

The wizard also facilitates the user to create dimensional hierarchies. Once the cube wizard is finished with the *dimensions* and *measures* selection it prompts the user for the cube storage mode as well. The *Storage Design Wizard* of MS Analysis Services allows for three types of storage modes, namely multidimensional (MOLAP), Relational (ROLAP) and Hybrid (HOLAP). Figure 9 shows the selection interface of storage type.

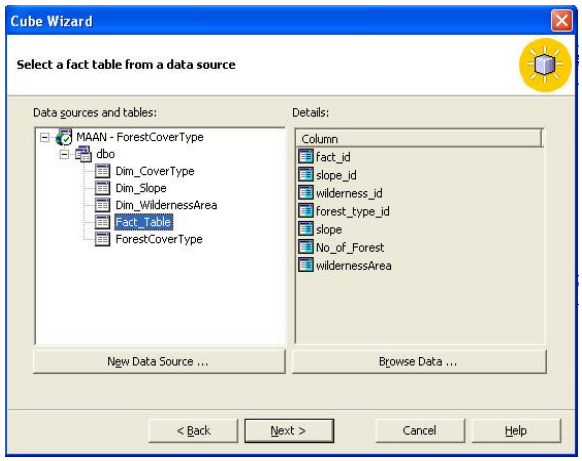


Figure 8. Fact Table Selection Step in the Cube Wizard.

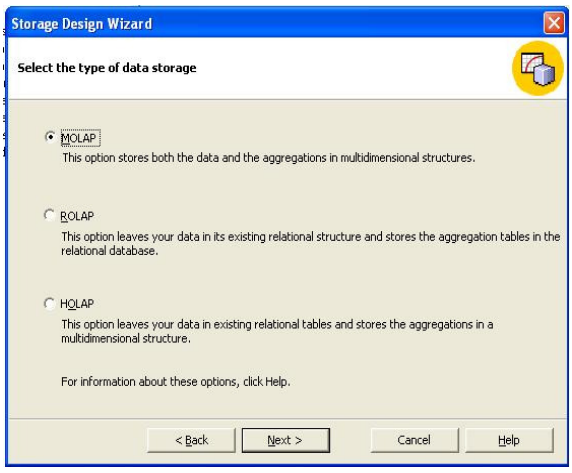


Figure 9. Storage Design Wizard – storage type selection step.

The cube performance can be controlled to some extent by setting the aggregation options of the cube. Aggregations are pre-calculated summaries of data that make querying of the cube faster. The wizard allows users to do setting of the aggregation options as well.

Figure 10 illustrates the aggregate setting option which can be controlled using both performance and size of the cube. Finally, the cube along with all the given settings, are processed and the details are shown to the user.

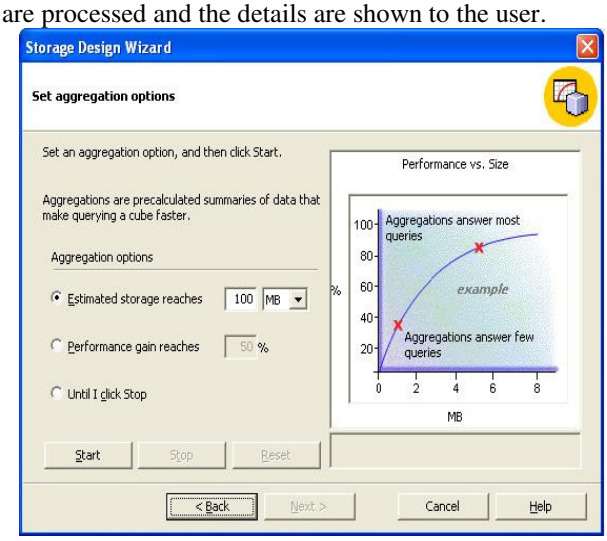


Figure 10. Cube Aggregation Settings –Storage Design Wizard.

Figure 11 shows the summary of the cube process. MS Analysis Services have a built-in *Cube browser* to view the cube data and perform the basic OLAP operations such as *drill-down* and *roll-up*.

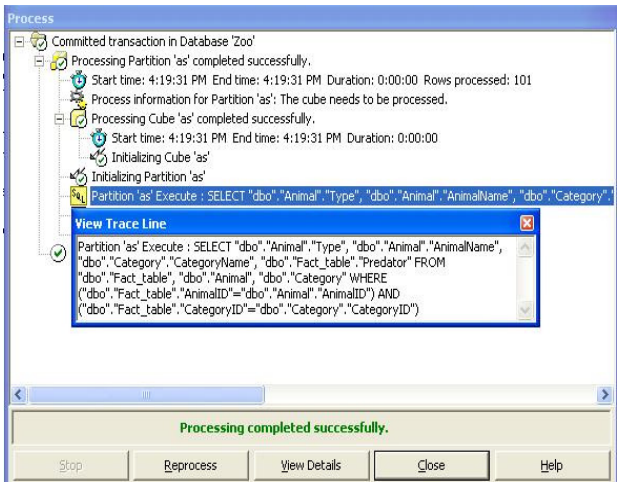


Figure 11. Animal Cube Construction Process Details.

Figure 12 presents a preview of the built-in cube browser of *MS Analysis Services*. The cube browser only shows data in a data grid. In order to enhance the visualization capabilities of the OLAP systems, we have developed a prototype using the *Dundas OLAP services.* With the aid of it, the prototype takes cubes residing in the OLAP server as a source and displays the cube data in the form of a rich and interactive interface. The user can perform the basic OLAP operations and can also see the data in a number of chart types and colorful grids. A number of reports can also be saved in Extensible Markup Language (XML) format.

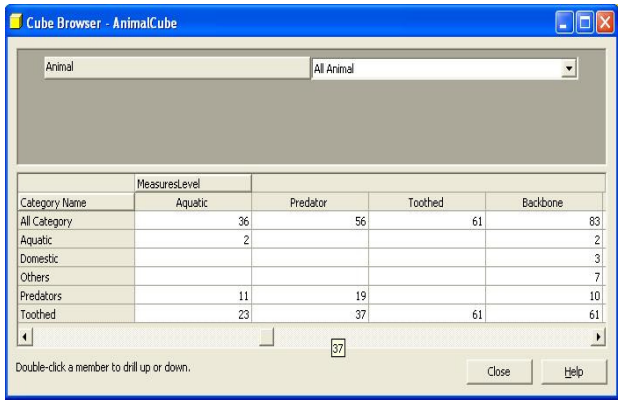


Figure 12. Animal Cube Data in Cube Browser.

Dundas OLAP Services has the following user interface controls: [30].

a) *OLAP Chart*: A central data visualization area to display the cube data.

b) *OLAP Grid*: A visualization grid for cube data-analysis.

c) *OLAP Toolbar*: Quick access to key control functions.

d) *Cube Selector*: A control to select a cube from a multi-cubed data source.

e) *Report List*: A collection of reports for storage purposes.
f) *Dimension Browser*: A control to browse and change the dimensions of the current cube.

Figure 13 shows the user interface of our prototype which has all the above mentioned controls to work on cube data. The prototype has been developed using *Microsoft visual studio 2005 and C sharp* programming langu*age*.

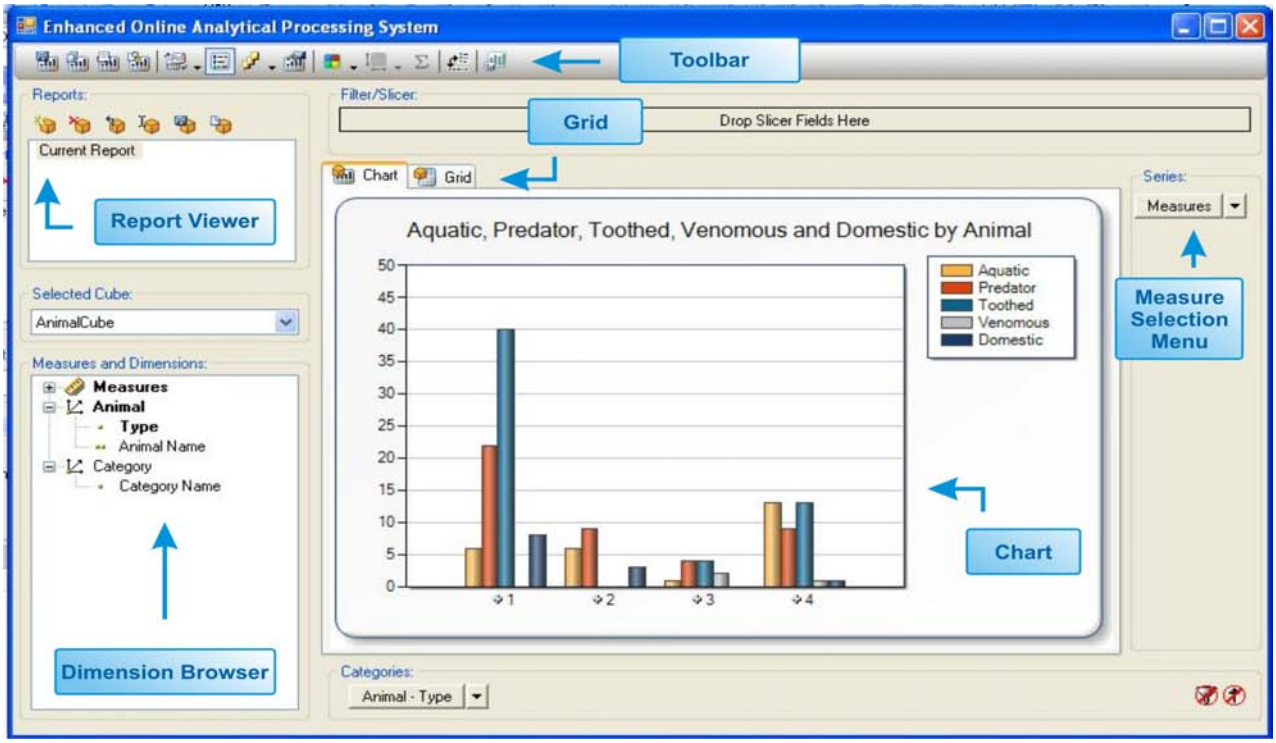


Figure 13. User Interface of Developed Prototype.

The second approach towards OLAP enhancement is the use of GSOM for loading the clustered data into the OLAP system. Data set is first fed into a GSOM tool which produces the hierarchical clusters using the spread factor. User can set the spread factor to control the number of hierarchical clusters generation using GSOM as depicted in Figure 4.

Once the clusters are generated the clusters are mapped manually into relational tables. The relational tables are stored in a database in the database server (MS SQL server 2000). From these relational tables star schema is created and uploaded with data. As mentioned previously the star schema becomes the data source and using this source, cubes of the clustered data are constructed. These cluster-based cubes become the source of data to be visualized using the prototype in the same manner.

Figure 14 shows the cluster-based cube data in the form of bar chart using chart view of the prototype. Using this approach, user can select the clusters of choice and perform analysis on particular clusters instead of the complete cube data.

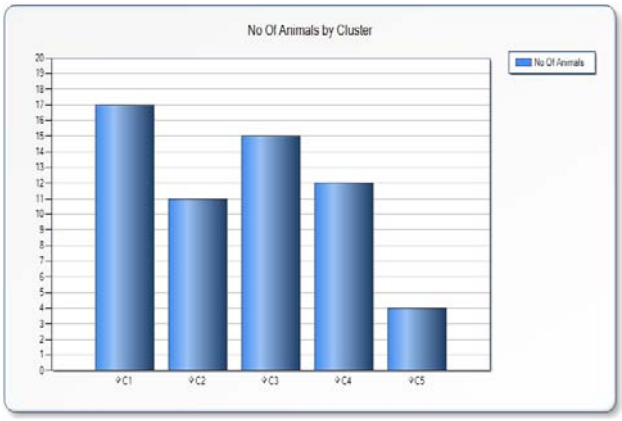


Figure 14. Chart view of clustered zoo data.

VI. EXPERIMENTS

As explained earlier, for the experiments we selected two different data sets for testing and validation of the results of our proposed architecture. The first is a limited dataset called the *Zoo* data set and the second one is a larger data set of *Forest Cover Type*. The idea is to verify how well our new architecture would perform for data of various sizes. Especially we want to demonstrate the integrated enhancements of performance and visualization. We tested both clustered and non-clustered

based approaches on the data sets. However, only the clustered based approach is discussed in details here.

A. Experiment 1 – with Simple Data

In our first experiment we chose a small data set called *Zoo* Data. Firstly the data set is passed to GSOM to generate the hierarchical clusters. We performed this process using different values of spread factor on the data in order to obtain hierarchical clusters at different levels of abstraction. Only three levels of hierarchical clusters were selected for analysis.

It was observed that the hierarchical clusters generated by using three different values of spread factor such as 0.01, 0.05 and 0.1 identify different groups and subgroups of several animal types. These groups clearly indicate the relevant grouping of data based on user interest and these clusters can be further spread out if necessary. GSOM provides an unbiased grouping of data in terms of clusters. These clusters were picked at diverse levels of abstraction and stored in relational tables. One of the main resulting tables is shown in Table 1.

TABLE I.
CLUSTER HIERARCHY TABLE FOR DATASET ZOO

ID	Clus_Name	Parent _Clus	Child _Clus	Child_ Child _Clus
1	Mammals	C1	C1-1	C1-1-1
2	Mammals	C1	C1-2	C1-2-1
3	Mammals	C1	C1-3	C1-3-1
4	Mammals	C1	C1-3	C1-3-2
5	Mammals	C1	C1-4	C1-4-1
6	Mammals	C1	C1-4	C1-4-2
7	Mammals	C1	C1-5	C1-5-1
8	Fish	C2	C2-1	C2-1-1
9	Fish	C2	C2-1	C2-1-2
10	Fish	C2	C2-2	C2-2-1
11	Bird	C3	C3-1	C3-1-1
12	Bird	C3	C3-2	C3-2-1
13	Bird	C3	C3-2	C3-2-2
14	Bird	C3	C3-2	C3-2-3
15	Crawler	C4	C4-1	C4-1-1
16	Crawler	C4	C4-1	C4-1-2
17	Crawler	C4	C4-2	C4-2-1
18	Crawler	C4	C4-3	C4-3-1
19	Insects	C5	C5-1	C5-1-1
20	Insects	C5	C5-1	C5-1-2

From these cluster relationship tables, we created a *fact table.* The two tables output from GSOM, namely Cluster Name Table and Cluster Hierarchy Table are used as *dimensions* to create a *star schema.* The fact table is created and updated using *SQL* queries as shown below.

- **Fact Table Creation**

```
CREATE TABLE [Fact_table] (  ZooID int
    identity(1,1) Not Null, [AnimalID] [int] NULL ,
    [CategoryID] [int] NULL, [Hair] [int] NULL,
    [Feathers] [int] NULL,  [Airborne] [int]
NULL,  [Aquatic] [int] NULL,  [Predator]
[int] NULL, [Toothed] [int] NULL,  [Backbone]
[int] NULL, [Breathes] [int] NULL,  [Venomous]
[int] NULL, [Fins] [int] NULL,  [Legs] [int]
NULL,  [Tail] [int] NULL,
[Domestic] [int] NULL,  [Catsize] [int] NULL
) ON [PRIMARY] GO
```

- **Fact Table Data Insertion**

```
INSERT INTO [Fact_table] (
[AnimalID],[CategoryID],[Hair],[Feathers],[Eggs],[Mil
    k],[Airborne],[Aquatic],[Predator],[Toothed]Backb
    one],[Breathes],[Venomous],
[Fins] ,[Legs],[Tail],[Domestic],[Catsize] )
```

- **Fact Table Data Update**

```
SELECT
    animalid,CategoryID,[Hair],[Feathers],[Eggs],[Mil
    k],[Airborne],[Aquatic],[Predator],[Toothed][Backb
    one],[Breathes],[Venomous],[Fins] ,[Legs] ,
        [Tail] ,[Domestic],[Catsize] FROM Animal
    a,Category c,Zoo_Data z WHERE
    a.AnimalName=z.Animal_Name and
    c.CategoryID=z.Category
```

TABLE II.
A FRAGMENT OF FACT TABLE OF CLUSTERED ZOO DATA

fact_id	Clus_id	no_of_animals
1	1	3
2	2	1
3	3	4
. . .		

The OLAP server takes this *star schema* and uses the clustered data to generate and process a cube. We have named the database having clustered zoo data as *Clustered_Zoo.* We further created an OLAP database and created a new cube called *cluster_zoo_cube.* The OLAP database which has stored the cube becomes the source of cube data. In the prototype, we set the connection so that the front-end tool can connect string with the OLAP server to get cube data and display it in the forms of charts, graphs, grids and reports. The connection string used to connect *Microsoft's Windows Forms* with the OLAP server is as follows;

Data Source=[server name]; Provider=msolap.2; Initial Catalog=Clustered Zoo

This connection string identifies the server name on which the OLAP server is hosted. Initial catalog is the OLAP database name which has the cubes stored in it. The prototype uses a function to display the cube data on the front end interface. The function used to display cube data on the client interface is as follow.

olapClient1.ShowData();

This function is responsible for showing the cube data so that the user can be facilitated with graph, charts and reports. Figure 16 shows the *cluster_zoo_cube data* using the prototype.

By using this interface, the user can drill down and roll up on the cube data and can see an instant manipulation of the chart. Switching from chart view to grid view is easy. Reports can be managed and saved using the report viewer. This interface offers an interactive visualization of charts. The basic operations that can be performed using this prototype are depicted in figures 17, 18 and 19. The bar charts show the hierarchy of the number of animals present in each cluster.

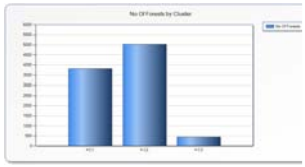


Figure 15. Chart View of Clustered Forest Data Cube.

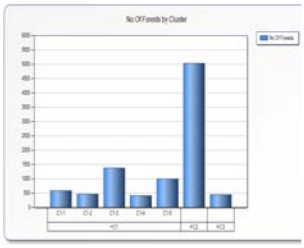


Figure 16. Drill Down on Cluster 1 (C1)

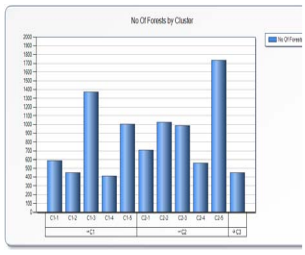


Figure 17. Drill Down on Cluster 2 (C2).

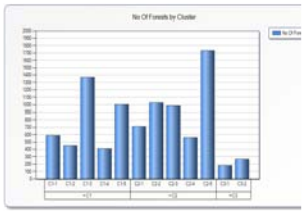


Figure 18. Drill Down on Cluster 3 (C3)

It is obvious from the prototype that the user can see the information of different clusters and can select any cluster of relevant interest for analysis. The presence of this unbiased grouping of data provides the user not only the selection of interested groups for OLAP operations but also to simplify processing of queries.

B. Experiment 2 – with Complex Data

The experiment was conducted over a large data set of *Forest Cover Type* [14]. The data set has 581,012 instances and 54 attributes. The attribute breakdown shows is has 12 measures with 54 columns of data from which 10 are quantitative variables, 4 are binary wilderness areas and 40 binary soil type variables. The data represent a typical analytic situation of certain complexity. The main purpose of the experiment is to validate the capabilities of our proposed architecture. We tested both clustered and non-clustered based approaches on the data sets. The class distribution of the Forest Cover Type data set is shown in table 3.

Firstly, the dataset is fed into the GSOM to generate hierarchical clusters. Hierarchical clusters were then transformed into relational tables. From the relational schema we created a Star Schema. An OLAP database called Clustered_ForestCovertype has been created to store cubes data. The star schema has one fact table and two dimension tables. Using this fact table and dimension tables we created a cube and named it as Clustered_Forest_Cube. This cube was stored in the OLAP database and that database was the source for the prototype. The cube was connected to the prototype and the cube data was shown on the front end. Readers are referred to [31] for more details of the process.

TABLE III.
CLASS DISTRIBUTION OF FOREST COVER TYPE [3]

Name	No. of Records
Spruce/Fir	211840
Lodgepole Pine	283301
Ponderosa Pine	35754
Cottonwood Willow	2747
Aspen	9493
Douglas-fir	17367
Krummholz	20510
Total	**581012**

The Fact table for Forest Cover type data set is in a similar format as in Table 1 in Experiment 1. Using this fact table and dimension tables we created a cube in the same way that was discussed in the previous experiment and named it as *Clustered_Forest_Cube*. This cube was stored in the OLAP database and that database became the source for the prototype. The cube was connected with the prototype and the cube data was shown on the front end. Figure 20 depicts the drill down and roll up operations performed on the clustered cube data showing the number of forests present in each cluster.

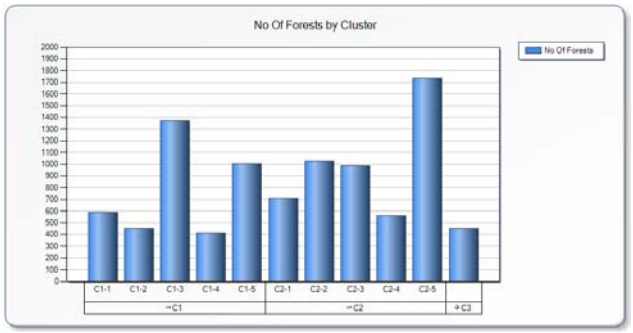


Figure 19. Samples of Drill-Down Operation on Clustered Forest Data.

It is clear from the prototype that the user can perform analysis on the cluster of his/her choice. The visually enriched interface allows interactive exploration of cube data hence enhances the power of OLAP system using the proposed architecture.

VII. DISCUSSION

The results of the experiments are discussed pertaining to the Forest Cover Type data set which represents an example of large size and complex data. This section is divided into two segments. In the first part we discuss the level of performance improvement achieved in terms of cube construction time. The second segment discusses the degree of visualization enhancement for the cube data

A. Performance Improvement

We have performed experiments for both clustered and non-clustered *Forest Cover Type* dataset. The non-clustered data set had originally 581,012 records. This huge amount of data has been loaded in a *MS SQL Server* database called *ForestCoverType*.

From this database a *star schema* has been generated using *SQL* queries, some of which are exposed in Figure 21. This schema became the source for the cube construction. From this non-clustered data set we generated a cube and named it *Forest Cube,* having 3 dimensions and 1 fact table.

```
insert into dbo.Fact_Table(slope_id,wilderness_id,forest  type_id,no_of_forest,
select
slope,Rawah_WA+Neota_WA+Comanche_Peak_WA+Cache_la_Poudre_WA,
cover_type,1 from ForestCoverType

update dbo.Fact_Table
set slope_id = 5
where slope_id >60 and slope_id <=75

update dbo.ForestCoverType
set Cache_la_Poudre_WA='4'
where Cache_la_Poudre_WA='1'
```

Figure 20. SQL Queries for star schema generation.

The *Forest Cube* process time was calculated and it was observed that it took 1 second to process 60,180 rows of data to construct the cube.

We carried out the same experiment on the clustered data which was first fed into the GSOM and then transformed into the S*tar Schema*. A cube named *Clustered_Forest_Cube* was generated using *Microsoft Analysis Services* depicted in Figure 22.

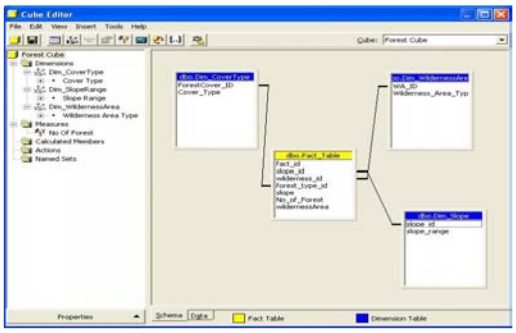


Figure 21. Star schema in Cube Editor of MS Analysis Services.

This clustered data cube was almost instantly processed and its construction time was less than a second. Figure 23 shows the cube construction time comparison of both clustered and non-clustered data.

The results of the experiments to calculate the construction time of a cube shows that the clustered data cube takes less time as compared to the non-clustered data cube. The significant thing about the graph is the rapid variation in the processing time. The processing time line of the non clustered data is increasing rapidly as the volume of data increases. In the case of clustered data the processing time does not increase so sharply. It is identified that if huge amount of data has to be dealt with in the construction of OLAP cube then the clustered approach is more suitable. For instance, the experiment performed clearly shows that if we increase the size of both clustered and non clustered data it affects the processing time of cube. Interestingly, the rate of increase of processing time when the data is not clustered is quite high. It can be seen in the graph that distance between the two lines keep on increasing as the data increases. For the clustered data the line remains below 200 units of processing time but it reaches up to 1000 units for the same non clustered data.

It is evident from the time comparison graph that the cube processing time can be reduced by using the clustered data, and the user can perform the targeted and fast multidimensional analysis on the clustered cube. Hence, it is shown from our benchmarking experiments that our proposed architecture yields performance improvement of OLAP data cube in computational time.

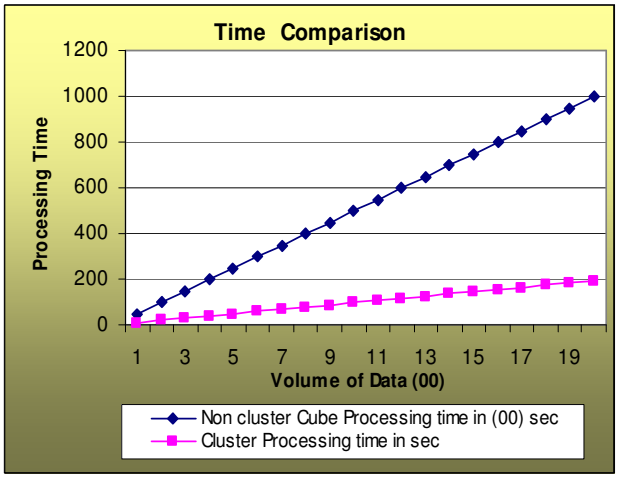


Figure 22. Cube Processing Time Graph.

B. Visualization Improvement

We have constructed cubes for both *Zoo* and *Forest Cover Type* data sets using *Microsoft's Analysis Services* OLAP engine. With the development of the prototype and embedding the OLAP data visualization controls by *Dundas Software,* we provide OLAP users with a rich user interface to perform targeted and interactive visual exploration of the data.

The user can view the same data in a number of charts and graphs simply by selecting the chart type from the *toolbar* in our prototype. For the sake of demonstration, we show the Clustered Zoo cube data using our prototype.

Figure 24 shows the outputs of the animal cube in various visual chart formats from our prototype.

Furthermore, users can perform an interactive analysis on the grid as well. Users can drill down and roll up to a level of detail using the "+" and "-" buttons present on the grid and charts. This interactive visualization is enhanced since the previous work only used the *GROUP BY* clause of *SQL queries* to retrieve data from the source system. The comparison of the previous data representation and the representation using the developed prototype is shown in Figure 25.

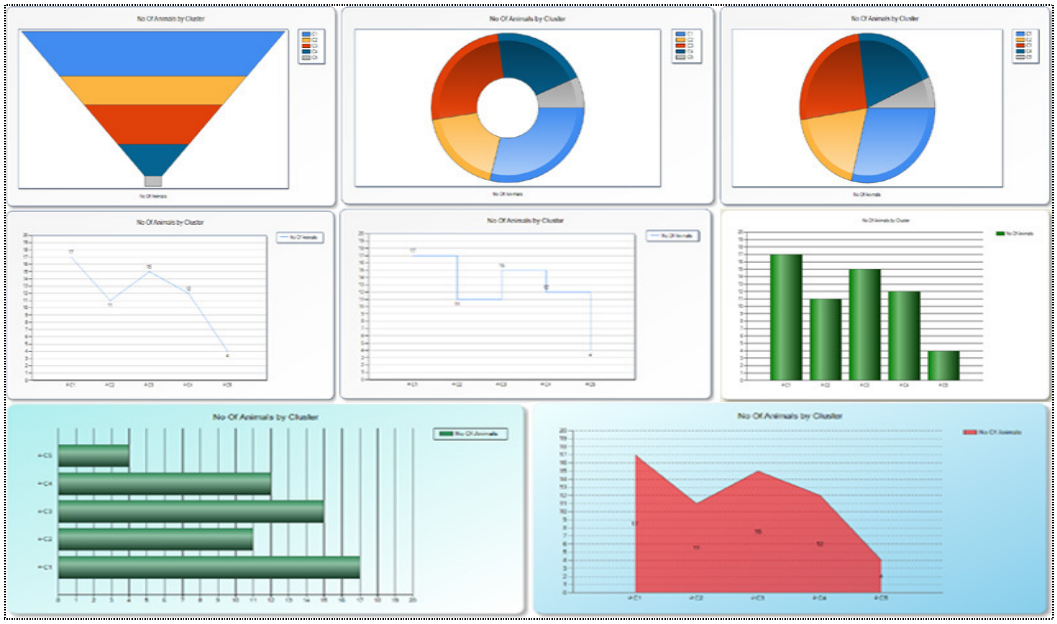
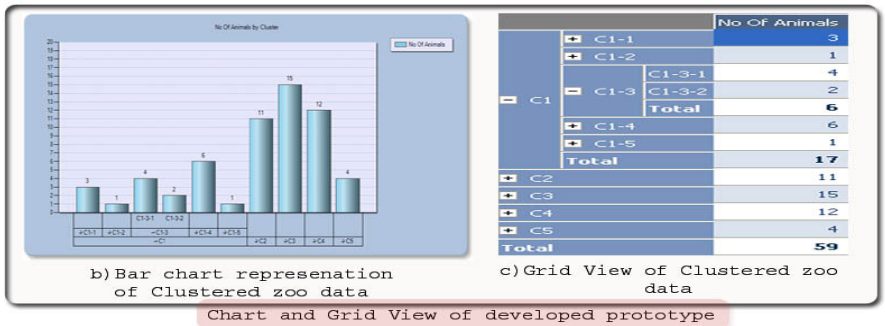


Figure 23. Preview of Different Chart Types Using Our Developed Prototype.

C1	C1-3	C1-3-2	By C1 By C1-3 By C1-3-2	By C1 By C1-3	By C1
Mammals	*Domestic Grass Eater*	*Family Caviidae*	2		
				6	
					8

a) Visual represetnation of cube data in previous work [6]



b) Bar chart represenation of Clustered zoo data

c) Grid View of Clustered zoo data

Chart and Grid View of developed prototype

Figure 24. Comparison of Visual Representation of Data by the Old and New Prototypes.

VIII. CONCLUSION AND FUTURE WORK

In this paper we proposed integrating OLAP performance and visualization enhancement techniques. To support this integration, we devised an architecture for the integrated enhancement using GSOM which generates hierarchical clusters as data input for OLAP. Hierarchical clusters help to enhance targeted analysis in OLAP by views of clusters which is not possible in relational database.

The proposed architecture along with its components is described in details. To demonstrate its advantages, a prototype has been developed and experiments are conducted over two real-life data-sets. It is observed that our architecture is relatively easy to implement and manage. Experimental results show that OLAP performance and visualization can be enhanced significantly. At the end we have compared the cube constructing times with those of the previous work, and emphasized the benefits of integrating performance improvement with visualization enhancement techniques.

Currently we are working on the dynamic generation of relational tables from the GSOM data. In addition to this we are also working on other Data Mining techniques that can be integrated with OLAP systems to enhance its analysis capabilities.

Furthermore, we are focusing on identifying the other limitations of the current OLAP systems. We are exploring how OLAP can be further extended and enhanced to meet the new challenges and to make it a more effective, efficient and intelligent OLAP system.

REFERENCES

[1] S. Mansmann and M. Scholl, “Exploring OLAP aggregates with hierarchical visualization techniques,” in *Proc. of ACM Symposium on Applied Computing*, 2007, pp. 1067-1073.

[2] S. Asghar and M. Usman, "Enhancing OLAP performance and visualization," in *Proc. of 6th Int'l Conf. on Information Science Technology and Management (CISTM)*, 2008, pp. 59-70.
[3] S. Asghar, D. Alahakoon and A. Hsu, "Enhancing OLAP functionality using self-organizing neural networks," *Neural, Parallel and Scientific Computations*, vol. 12, no. 1, pp. 1-20, March 2004
[4] S. Chaudhuri and U. Dayal, "An overview of data warehousing and OLAP technology," *ACM SIGMOD Record*, vol. 26, no. 1, pp. 65-74, March 1997.
[5] S. Goil and A. Choudhary, "High performance OLAP and data mining on parallel computers," *Data Mining and Knowledge Discovery*, vol. 1, no. 4, pp. 391-417, Dec. 1997.
[6] J. Hipp, U. Guentzer and G. Nakhaeizadeh, "Algorithms for association ruling mining – a general survey and comparison," *ACM SIGKDD Explorations Newsletter,* vol. 2, no. 1, pp. 58 – 64, June 2000.
[7] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine Learning,* vol. 11, no. 1, pp. 63-90, April 1993.
[8] A. K. Jain, M. N. Murty and P. J. Flynn, "Data clustering: A review," *ACM Computing surveys (CSUR),* vol. 31, no. 3, pp. 264-323, Sep. 1999.
[9] M. J. Shaw, C. Subramaniam, G. W. Tan and M. E. Welge, "Knowledge management and data mining for marketing," *Decision Support Systems,* vol. 31, no. 1, pp. 127-137, May 2001.
[10] M. Sifer, "A visual interface technique for exploring OLAP data with coordinated dimension hierarchies," in *Proc. of 12th ACM Int'l Conf. on Information and Knowledge Management (CIKM)*, 2003, pp. 532-535.
[11] C. Siedman, *Data Mining with Microsoft SQL Server 2000 Technical Reference,* Microsoft Press, Redmond, WA, 2001.
[12] S. Soni and W. Kurtz, "Analysis services: Optimizing cube performance using Microsoft SQL server 2000 analysis services," *Microsoft SQL Server 2000 Technical Articles* March 2001.
[13] "Dundas Chart for ASP.NET," Help Document, [Online], http://support.dundas.com/OnlineDocumentation/WebChart2005.
[14] J. A. Blackard, D. J. Dean and C. W. Anderson, Forest cover type, *The UCI KDD Archive* [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science (1998).
[15] E. Keogh, C. Blake and C. J. Merz, "UCI repository of machine learning database," 1999. [Online]. http: / /www.ics.uci.edu /~mlearn /MLRepository.html
[16] T. Imielinski, A. Virmani and A. Abdulghani, "DataMine: Application programming interface and query language for database mining," in *Proc. of 2nd KDD Conf.,* 1996, pp. 256-262.
[17] D. Alahakoon, S. K. Halgamuge and B. Srinivasan, "Dynamic self organising maps with controlled growth for knowledge discovery," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 601-614, May 2000.
[18] T. Kohonen, *Self-Organising Maps,* 3rd ed. Springer-Verlag, Berlin, 2001.
[19] V. Markl, F. Ramasak and R. Bayer, "Improving OLAP performance by multi-dimensional hierarchical clustering," in *Proc. of 1999 Int'l Symposium on Database Engineering and Applications*, 1999, p. 165.
[20] D. Theodoratos and A. Tsois, "Heuristic optimization of OLAP queries in multidimensionally hierarchically clustered databases," in *Proc. of 4th ACM Int'l Workshop on Data Warehousing and OLAP,* 2001, pp. 48-55.
[21] R. B. Messaoud, O. Boussaid and S. Rabaseda, "A new OLAP aggregation based on the AHC technique," in *Proc. of 7th ACM Int'l Workshop on Data Warehousing and OLAP (DOLAP)*, 2004, pp. 65-72.
[22] A. S. Maniatis, P. Vassiliadis, S. Skiadopoulos and Y. Vassiliou, "Advanced visualization for OLAP", in *Proc. of 6th ACM Int'l Workshop on Data Warehousing and OLAP (DOLAP)*, 2003, pp. 9-16.
[23] A. Voss, V. Hernandez, H. Voss and S. Scheider, "Interactive visual exploration of multidimensional data: Requirements for common GIS with OLAP," in *Proc. of 15th Int'l Workshop on Database and Expert Systems Applications (DEXA),* 2004, pp. 883-887.
[24] M. Scotch and B. Paramanto, "SOVAT: Spatial OLAP visualization and analysis tool," in *Proc. of 38th Annual Hawaii Int'l Conf. on Systems Sciences (HICSS),* 2005, p. 165.
[25] A. Cuzzocrea, D. Sacca and P. Serafino, "A hierarchy driven compression technique for advanced OLAP visualization of multidimensional data cubes", *in Proc. of 8th Int'l Conf. on Data Warehousing and Knowledge Discovery (DaWak'06), Springer Verlag* 2006*,* pp. 106-119.
[26] K. Techapichetvanich and A. Datta, "Interactive visualization for OALP," in *Int'l Conf. on Computational Science and its Applications (ICCSA),* 2005, pp. 206-214.
[27] S. Mansmann and M.Scholl, "Extending visual OLAP for handling irregular dimensional hierarchies," in *Proc. of 8th Int'l Conf. on Data Warehousing and Knowledge Discovery (DaWaK'06), Springer Verlag* 2006, pp. 95-105.
[28] S. Mansmann and M. Scholl, "Visual OLAP: A new paradigm for exploring multidimensional aggregates," in *Proc. of IADIS Int'l Conf. on Computer Graphics and Visualization (CGV),* 2008, pp. 59-66.
[29] B. Fritzke, "Growing cell structures - a self organizing network for unsupervised and supervised learning," *Neural Networks,* vol. 7, no. 9, pp. 1441-1460, 1994.
[30] Dundas Chart for ASP.NET - OLAP Services, Overview of Dundas OLAP Architecture, 2005 - 2009 Dundas Data Visualization, Source:http://support.dundas.com/OnlineDocumentation/WebOLAP/Overview%20of%20Dundas%20OLAP%20Architecture.html. [Online].[Accessed: Dec. 2008].
[31] M. Usman, S. Asghar, S. Fong, "An Architecture of Integrated Enhancement of OLAP Using Growing Self Organizing Neural Networks", *International Conference on Computer Engineering and Systems, Cairo, Egypt*, IEEE, submitted.