QoS: Quality Driven Data Abstraction for Large Databases

by

Charudatta Wad

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

February 2008

APPROVED:

Professor Elke A. Rundensteiner, Thesis Advisor

Professor Murali Mani, Thesis Reader

Professor Michael A. Gennert, Head of Department

Abstract

Data abstraction is the process of reducing a large dataset into one of moderate size, while maintaining dominant characteristics of the original dataset. Data abstraction quality refers to the degree by which the abstraction represents original data. Clearly, the quality of an abstraction directly affects the confidence an analyst can have in results derived from such abstracted views about the actual data. While some initial measures to quantify the quality of abstraction have been proposed, they currently can only be used as an after thought. While an analyst can be made aware of the quality of the data he works with, he cannot control the desired quality and the trade off between the size of the abstraction and its quality. While some analysts require atleast a certain minimal level of quality, others must be able to work with certain sized abstraction due to resource limitations. consider the quality of the data while generating an abstraction. To tackle these problems, we propose a new data abstraction generation model, called the QoS model, that presents the performance quality trade-off to the analyst and considers that quality of the data while generating an abstraction. As the next step, it generates abstraction based on the desired level of quality versus time as indicated by the analyst. The framework has been integrated into XmdvTool, a freeware multi-variate data visualization tool developed at WPI. Our experimental results show that our approach provides better quality with the same resource usage compared to existing abstraction techniques.

Acknowledgements

I would like to take this oppurtunity to thank my parents Mrs. Pournima Wad and Mr. V. M. Wad. Their encouragement and support has always been my biggest strength. I would also like to thank my sister Minal Wad for her guidance. I would also like thank my friend Prachi Gupta for her constant support.

I would like to express my greatest gratitude to my advisors, Prof. Elke Rundensteiner and Prof. Matthew Ward, for their guidance and invaluable contributions to this work. Prof. Elke Rundensteiner has been a friend, motivational guru and an exceptional teacher.

I would like to thank Prof. Murali Mani for being the reader of this thesis and giving me much valuable feedback.

I would like to thank XMDV team members, Zaixian Xie, Quingguang Cui, Di Yang and Do Quyen Nguyen. I would also like to thank all the members of DSRG for their valuable feedback.

I would also like to thank Jing Yang, whose dimensional InterRing code was used.

Finally, I would like to dedicate this work to my gramdmother Mrs. K. M. Wad whose blessings are always with me.

Contents

List of Figures

Chapter 1

Introduction

1.1 Multi-dimensional Visualizations

The amount of data generated due to technological advances is enormous. Most data generated and collected is a valuable source of information. However, finding the information from the huge amount of data is a difficult task. One approach is to present the human analyst the data in graphical form. This allows the analyst to apply his perceptions to make sense of the data and derive conclusions. This approach is called ”data visualization”.

Multivariate visualization is one subfield of data visualization that focusses on multi-dimensional datasets. A multi-dimensional dataset can be defined as a set of data items D, where the i^{th} data item d_i consists of a vector with n variables, (x_{i1}, x_{i2}, ..., x_{in}). Each variable may be independent of or interdependent with one or more of the other variables. Variables may be discrete or continuous in nature, or take on symbolic (nominal) values.

Many multivariate visualization techniques and systems have emerged during the last three decades, such as glyph techniques [1, 2, 3, 4], parallel coordinates [5, 6], scatterplot matrices [7], pixel-level visualization [8], and dimensional stacking [9]. Each method has strengths and weaknesses in terms of the data characteristics and analysis tasks for which it is best suited.

Recently lots' of attention has been focussed on visualization of large high-dimensional datasets. Though there are various methods available for abstracting the large datasets and visualizing them. There is no way to validate or more importantly to control the quality of abstraction that is generated.

1.2 Motivating Example

Data abstraction techniques are commonly used to facilitate the efficient detection of patterns in large datasets and for analyzing a huge database without having to actually explore the original data [10]. Thus, analysts typically infer characteristics of large databases by analyzing the abstracted dataset and rather than looking at the full data. Some abstraction techniques select a subset of the original dataset as its abstraction, such as sampling and filtering, while others construct a new abstract/summary representation, such as clustering and summarizing [10]. Tasks conducted based on abstracted data include pattern detection, cluster analysis, outlier analysis, subspace cluster analysis, filtering and sample analysis [10]. Figure 1.1 presents an example of a dataset and its abstractions. The visualization technique used, called parallel coordinates [11], is a popular multivariate visualization technique.

Since the analyst may rely on the abstraction visualization to derive conclusions, the quality of the abstraction needs to be validated. A good data abstraction represents all the main features of the original dataset. Since the abstraction in Figure 1.3 captures all the clusters present in the original dataset (Figure 1.3(a)) abstraction is considered to be of high quality. Whereas the abstraction in the Figure 1.3 missed out on a small cluster, thus the abstraction can be considered to be of low quality. Lack of knowledge regarding quality of the data one works with can lead to inaccurate results jeopardizing the reliability

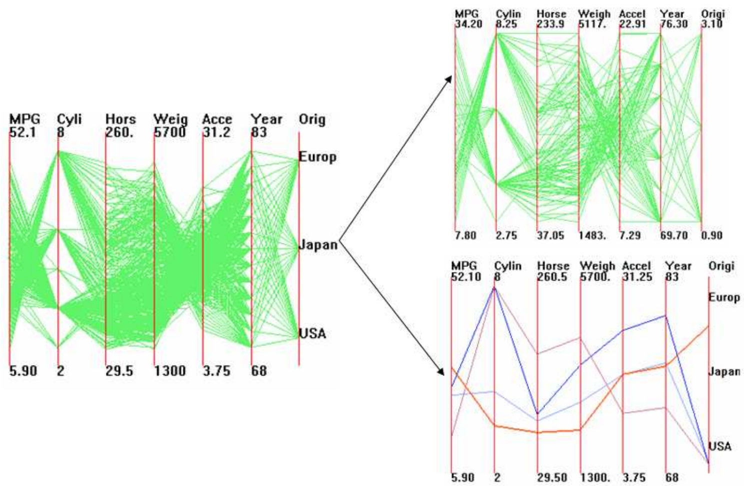

Figure 1.1: Figure on the left displays the cars dataset, while Figures on the right represents sampling of the cars dataset and cluster centers of cars dataset.

of conclusions gleaned from the abstraction. Validating the quality of abstraction is made difficult due to lack of data abstraction quality measures. Although some initial measures [12] have been recently proposed to measure the data abstraction, those measures do not scale well to higher number of dimensions. Furthermore, a scalable data abstraction measure by itself does not solve the problem. The main problem is the lack of consideration about the desired quality by the abstraction generation process before its commencement.

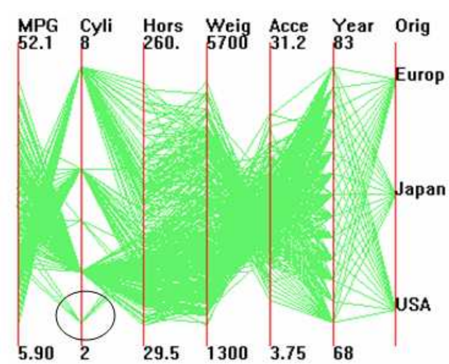

Figure 1.2: Visualization of the original dataset cars.

To further complicate matters, we note that most systems and thus users of these systems assume that the raw data itself is always good. However, real-world data is

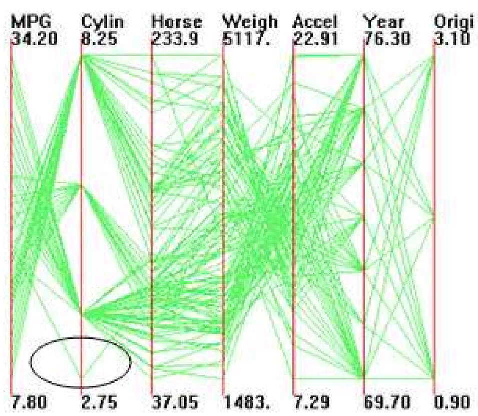

Figure 1.3: Abstraction of dataset cars capturing all clusters.

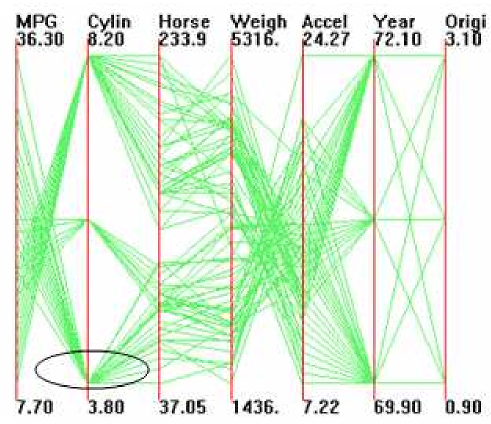

Figure 1.4: Abstraction of dataset cars missing out on a small cluster.

known to be imperfect, suffering from various forms of defects such as sensor variability, estimation errors, uncertainty, human errors in data entry, and gaps in data gathering. *Data quality* refers to the quality of the underlying data used for the abstraction generation. If the quality of the underlying data is not considered during abstraction generation, the quality of an abstraction is endangered.

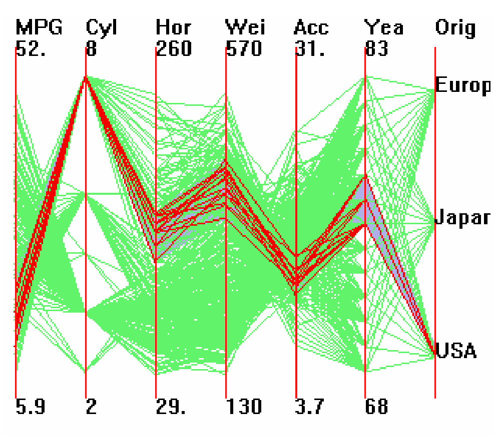

Figure 1.5: Low quality data in the original dataset.

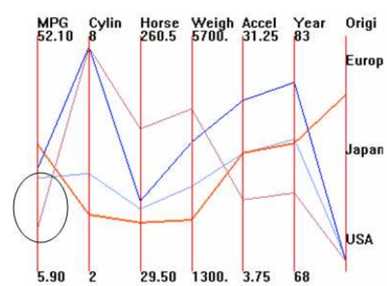

Figure 1.6: Effect of low quality data on abstraction visualization.

Consider the Figure 1.5, the cluster marked is of low quality. If the data quality of the underlying data is ignored during the abstraction generation, the resulting clusters (Figure 1.6)) may have good abstraction quality. The quality of abstraction also depends on the underlying data quality of the raw data used for clustering. Thus, data quality should be considered during abstraction generation to calculate a total abstraction quality.

1.3 Problem Definition

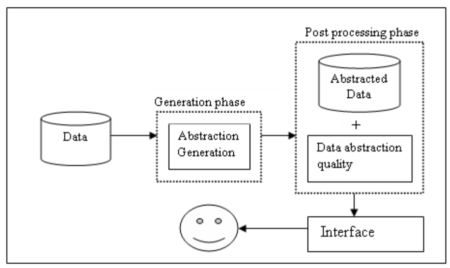

Figure 1.7: Process Flow Existing Data Abstraction Solution.

Figure 3.1 sketches the process most commonly used by abstraction generation systems [13] [14]. Predicaments of such a process include:

- Quality measures, if available at all, are plugged in only as an after-thought to calculate the quality of a given abstraction.

- Data abstraction is a one way process. Thus, when an analyst generates an abstraction, he in unaware of the abstraction quality until after the abstraction process has been completed. In some sense he blindly completes the abstraction process and hopes for the best that the quality abstraction will be acceptable.

- Furthermore, the analyst doesn’t know how much time they should expect to have to wait for the completion of the abstraction process. He cannot schedule a process based on the time or task at hand.

- Data quality is not taken into account. As discussed earlier, if the data is imperfect (or of low quality), the abstraction result should also reflect the underlying data quality.

- Analyst cannot visualize the quality of abstraction.

1.4 Goals of the Thesis

In this thesis, we intend to explore a new approach to quality based data abstraction generation and visualization. It allows the user to control the quality of abstraction to be generated, allowing him to interact with the system and setting either performance or quality value and QoS will generate the abstraction based on the settings. Goals of QoS are:

- to present the analyst with a quality-performance trade off indicating the different values of quality measures achievable and time required for the process to generate them. Using these computations, analyst can demand a quality level beforehand or he can request a certain performance, knowing what quality he can expect and QoS will generate the abstraction accordingly.

- to take into consideration both the data abstraction quality and underlying data quality to calculate a complete data abstraction quality measure.

- to provide a way to visualize the abstraction quality using an InterRing display

1.5 Approach in a Nut Shell

To overcome the above identified problems, we propose to make abstraction generation quality aware by adding a new layer of estimation. The approach can be divided into four steps:

1. Density Estimation: In this step, we estimate the density of the dataset by creating a multi-dimensional histogram. The multi-dimensional histogram keeps a track of number of bins formed and data points falling into each bin.

2. Formation of look-up table: Using density-biased sampling we calculate the data abstraction measure for various sampling levels and time required for the clustering process to complete.

3. Abstraction Generation: Once the analyst selects a particular quality value, abstraction is generated. Data quality of the clusters is calculated and total abstraction quality is calculated for the clusters.

4. Cluster Visualization: Clusters formed are visualized using the InterRing display.

1.6 Thesis Organization

Recent research regarding data abstraction for large scale datasets, different forms quality, cluster visualization and other quality aware systems are surveyed in Chapter 2. Chapter 3 presents details of the QoS approach. Chapter 4 desribes the inclusion of data qualit in QoS. Visualizing and navigation through the clustering result is explained in Chapter 5. Chapter 6 presents our implementation in the XmdvTool. We discuss our ecperimental evaluation in Chpater 7. Conclusions and open questions are discussed in Chapter 8.

Chapter 2

Background

2.1 Data Abstraction for Large Scale Explorations

There are many approaches towards visualizing large-scale multi-dimensional data sets, such as pixel-oriented techniques (including spirals [15], recursive patterns [8], and circle segments [16]), multiresolution multidimensional wavelets [17], pixel bar charts [18], and interactive hierarchical displays [19, 20, 21].

Hierarchical Parallel Coordinates [22] is one of the interactive hierarchical displays [21] developed for visualizing large multidimensional data sets in the context of our XMDV project. Since displaying a large number of data items will clutter the screen, Hierarchical Parallel Coordinates group the data items into a hierarchical cluster tree. A set of clusters selected from a certain level of detail in the hierarchical cluster tree is visualized on the screen instead of all the data items in the data set. The clusters are visualized by center lines and bands which respectively represent the mean points and extents of the clusters. The same framework used to develop Hierarchical Parallel Coordinates has been also applied to Hierarchical Scatterplot Matrices, Hierarchical Star Glyphs, and Hierarchical Dimensional Stacking. Details of these can be found in [20].

Hierarchical Parallel Coordinates uses clustering to obtain the hierarchical cluster tree. However, other abstraction techniques such as sampling or clustering can also be used. Sampling and clustering are well studied in literature.

2.1.1 Sampling

Sampling is a form of abstraction where the original data points are used to create an abstraction. Sampling has been extensively studied. Different methods of sampling are:

- Simple random sampling: Random sampling implies that every data point has equal probability of being selected in the sample [23]. Random sampling has an advantage in terms of its simplicity and ease of implementation. However, random sampling has a disadvantage in terms of not including data points from small cluster. Olken et. al. [24] introduced a sampling operator into DBMS with the goal to increase efficiency. By embedding the sampling within the query evaluation, one can reduce the amount of data which must be retrieved in order to answer sampling queries. Sampling can be used in the DBMS to provide cheap estimates of the answers of aggregate queries. Sampling may also be used to estimate database parameters used by the query optimizer to choose query evaluation plans. Olken et. al. [24] then introduced the idea of weighted random sampling through sampling from B+ tree, hash files and spatial data structures (including R-trees and quadtrees)).

- Stratified random sampling: Stratification is the process of grouping members of the population into relatively homogeneous subgroups or ”strata” before sampling. Strata should be mutually exclusive and collectively exhaustive [23]. Once strata are formed, random sample is chosen from each strata. The samples are then combined to form the overall samples. For the formation of strata the input data should be discrete. It is not usefult when there are no homogeneous subgroups and it can be difficult to find and select relevent stratification variables in presence of homogeneous subgroups.

- Density biased sampling: Density biased sampling is a type of probability based sampling where probability is assigned to a group of data [25]. Suppose that we have N values x_1, x_2,... ,x_N that are partitioned into g groups that have sizes n_1,n_2,..., n_g. Suppose we want to generate a sample with expected size M in which the probability of point x_i being in the sample is dependent on the size of the group containing x_i. The density biased sample has the following properties [25]:

  – Within a group, points are selected uniformly.

  – The sample is density preserving.

  – The sample is biased by group size.

  Thus, density biased sampling preserves the density and captures even the small clusters present in the dataset.

Random sampling vs. Density biased sampling: For the purpose of QoS, we need to estimate the quality values before sampling. Every quality value is calculated for a particular sampling rate. With random sampling, for a particular sampling rate there can be various quality values possible. Whereas with density biased sampling, a quality value is associated with only one sampling rate.

2.1.2 Clustering Algorithm

There are two basic types of clustering algorithms [26, 27] partitioning [28, 29] and hierarchical algorithms [30, 31, 32]. Partitioning algorithms divide all the data points into a given number of clusters, while hierarchical algorithms construct a hierarchical cluster tree by recursively splitting the data set into smaller clusters until every leaf cluster contains only one or a few data points.

The k-means algorithm is a popular partitioning algorithm. It picks k cluster centroids and assigns points to the clusters by picking the closest centroid to the point in question. The centroids of the clusters may shift when new points are added into clusters so the process may need to be repeated. BFR [29] is an algorithm based on k-Means algorithm. It intends to cluster large data sets that cannot be loaded into the main memory at one time by identifying regions of the data that are compressible, regions that must be maintained in memory, and regions that are discardable. This algorithm works best if the clusters are normally distributed around some central points.

CURE [30] is a sampling-based hierarchical clustering algorithm for large data set. Compared with k-means approaches, which work well only for clusters that are neatly expressed as Gaussian noise around a central point, CURE is more robust in that it is able to identify clusters having non-spherical shapes and wide variances in size. CURE achieves this by representing each cluster by a certain fixed number of points that are generated by selecting well scattered points from the cluster and then shrinking them toward the center of the cluster by a specified fraction. Having more than one representative point per cluster allows CURE to adjust well to the geometry of non-spherical shapes and the shrinking helps to dampen the effects of outliers. To handle large databases, CURE employs a combination of random sampling and partitioning. A random sample drawn from the data set is first partitioned and each partition is partially clustered. The partial clusters are then clustered in a second pass to yield the desired clusters.

BIRCH [32] is another efficient clustering algorithm for large data sets. In the BIRCH algorithm, objects are read from the database sequentially and inserted into incrementally evolving clusters that are represented by generalized cluster features (CFs). A new object read from the database is inserted into the closest cluster, an operation which potentially requires an examination of all existing CFs. Therefore BIRCH organizes all clusters in an in-memory index, a height-balanced tree called a CF-tree. For a new object, the search for an appropriate cluster now requires time logarithmic in the number of clusters as opposed to a linear scan.

For high dimensional space, it is common that clusters only exist in some subspaces. CLIQUE [33] is a clustering algorithm that is able to find clusters embedded in subspaces of high dimensional data. It identifies dense clusters in subspaces of maximum dimensionality. It generates cluster descriptions in the form of DNF expressions that are minimized for ease of comprehension.

Human interaction in the field of clustering was first introduced by K. Chen et. al. [34] via the VISTA software. Using VISTA, the user is able to participate in the clustering process by steering,monitoring or refining the clustering process. VISTA allows the user to improve the clustering process by introducing their domain knowledge in the clustering process. However, VISTA deals with small and moderate datasets.

In this thesis, we tend to use the existing abstraction techniqu of clustering and make them quality aware.

2.2 Forms of Qualities and Quality Measures

2.2.1 Abstraction Quality

Abstraction quality captures how well the abstracted dataset represents the original dataset. It quantifies to what degree the abstraction is an appropriate representation of the original dataset. As discussed earlier, if the abstraction misses out on clusters present in the original dataset, then the abstraction is said to be of low quality. Recently some abstraction measures are introduced in the field of information visualization. Cui et. al. [12] proposed a histogram based measure. Our measure is an extension to HDM [12] capturing the co-relationships between the dimensions using a multi-dimensional histogram. Details on our quality measure MHDM are described in Chapter 3.

2.2.2 Data Quality

Data Quality [35] denotes the degree of uncertainty about the data. High quality indicates that data is of high certainty and reliability. The variability of data quality has many causes such as data accuracy, completeness, certainty, consistency, or any combination of these. It can also include statistical variations or spread, errors and differences, minimum-maximum range values, noise, or missing data [36]. Calculation of data quality is out of scope of this thesis. More details about calculation of data quality can be found at [36].

In the Xmdv project, we employ scalar values to measure uncertainty. The quality is captured at three granularity namely individual data values, complete records and specific dimensions.

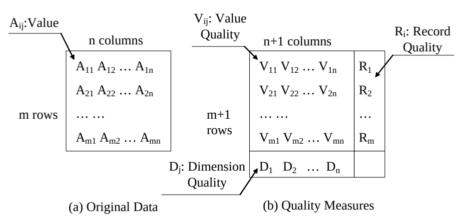

Figure 2.1: Structure of data quality.

Figure 2.1 represents the data quality structure defined and used in Xmdv. Quality at individual data values is summarized at the record level. Since, clustering is performed at the record level, we use the summarized record quality value instead of individual data values and the specific dimensional values. Thus, we only ise the record quality values and ignore the dimensional and individual data quality values. Dimensional quality values can be useful in dimensional clustering. The discussion of integrating dimensional quality values with dimensional clustering is out of the scope of this thesis.

2.2.3 Cluster Quality

Abstraction quality and cluster quality refers to the quality of the hierarchical cluster tree. Thus, we refer to abstraction quality and cluster quality as ”structure quality” depicting the quality of the structure formed as a result of clustering. Clustering quality is the evaluation of the results of clustering algorithm. One of the criteria used to measure the clustering quality is the compactness of the cluster. Compactness can be defined as [14] the closeness amongst the members of the cluster. Variance can be used as a measure of caompactness. Due to its simplicity, we use variance as our clustering quality measure. Any other measure such seperation of the cluster [14] can also be used. We discuss our clustering quality measure in greater detail in Chapter 3.

2.3 Cluster Visualization

Recent literature indicates that radial space-filling techniques work better in revealing hierarchical structures than treemaps [37, 38], while also making efficient use of the display space. Sunburst [39] is an example of the radial space-filling hierarchy visualization technique. In Sunburst, deeper nodes of the hierarchy are drawn further from the center and child nodes are drawn within the arc subtended by their parents. The angle occupied by a node is proportional to its size.

Radial space-filling techniques have some advantages over other tree drawing strategies. First, as one of the space-filling techniques, they use more implicit containment and geometry characteristics to present a hierarchy than tree drawing algorithms. The later utilize edges between nodes to indicate parent-child structure [39]. Second, compared to treemaps, radial space-filling techniques are better in conveying the hierarchical structure [40, 41, 39]. However, radial space-filling techiniques have a drwback, the small slices of the clustering result are difficult to distinguish. This shortcoming in overcome by using context+focus [21].

2.4 Data Visualization vs. Quality Visualization

Visualization of data has been an important research topic for years. Data visualization implies visualizing the data for analysis. Various displays such as Parallel Coordinates, Scatterplot matrices etc. are used to visualize and analyze the data. However, the validity of the decisions made and information extracted by the exploratory data visualization largely depends on the quality of the data. Quality of the data here refers to the several differen types of qualities as discussed earlier. Therefore, visualization of the quality along with the data has been identified as a critical research topic in recent years. This creates two types of navigation namely navigation through the data space and quality space. Navigation in the quality space refers to querying the quality of the data. However, quality space and data space are interrelated in terms of navigation in one space provides results in both the data and quality space.

In the recent literature[35], there are number of research activities focussed on visualization of quality attributes of the data. XmdvTool research group has come up with visualizing the data quality in the multi variate data visualization. However, the work is limited to visualizing the data quality. We plan to extend the work by conveying the structure quality through visualization.

2.5 Other Quality Aware Systems

Widom et. al. proposed a data model called Trio system [42] which incorporated lineage and accuracy of the data. Trio project combines and distills the existing DBMS to include accuracy and lineage through an extension to SQL language. Trio provides querying in three interrelated components namely, data, accuracy and lineage. Accuracy of the data implies the amount of confidence or uncertainty of the data, whereas lineage of the data is the accuracy of the derivation. Trio system maintains the lineage of the data as it is derived in the database allowing the queries to specify the expected data quality and lineage. Trio system allows querying in the data as well as quality space. However, Trio system does not deal with quality of abstracted data, nor with clustering. Trio system maintains the quality of the data but there is no provision to improve or produce the quality that user specifies.

Chapter 3

QoS: Quality Driven Abstraction

3.1 QoS Framework

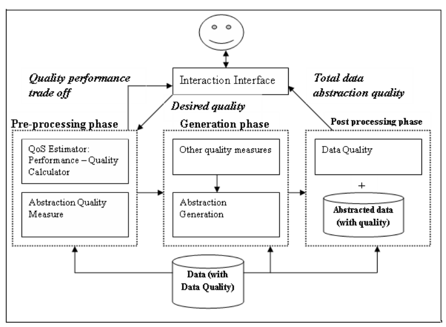

Figure 3.1: QoS Framework.

The system framework for QoS, depicted in Figure 3.1, consists of the following main phases:

1. *Pre-processing phase:* We introduce a pre-processing phase to compute the quality-performance trade-off. The computation is done using a multi-dimensional histogram which calculates density information. Two main components in this phase are:

1) A scalable data abstraction measure to quantify the data abstraction result is proposed, called Multi-dimensional Histogram Difference Measure (MHDM). Other measures [12] could also be plugged in.

2) The estimator calculates the performance-quality trade-off including confidence intervals and time estimations for the process. This is at the heart of QoS, presenting the analyst with various trade-offs before the process of abstraction commences.

2. *Generation phase:* This process generates an abstraction based on quality values set by the analyst.

3. *Post-processing phase:* It combines the measure of the abstraction with quality of the underlying dataset to determine the overall quality.

4. *Interaction interface:* This interface presents the performance quality trade off and the final abstraction quality to the analyst.

3.2 QoS for Cluster Analysis

Summarization techniques for data abstraction summarize the data by creating a small number of representatives to represent the underlying larger volume of data [10]. Clustering is one such technique where cluster representatives are used to represent the data. Since clustering is memory and computationally intensive, clustering of large databases typically employs sampling as an pre-processing step for clustering [32][30]. Sampling reduces the number of points to be clustered, thus making clustering computationally practical for large databases.

3.2.1 Quality Measure: MHDM

We now propose a measure of the data abstraction quality for high dimensional data. The measure can be calculated before the abstraction is actually generated. This multi-dimensional data abstraction quality measure which captures the co-relationships present in a high dimensional dataset. The proposed measure, called Multi-dimensional Histogram Difference Measure (MHDM), is a histogram difference method. Histograms are widely used for density and selectivity estimation [43]. It calculates the difference between the multi-dimensional histogram of the original dataset and that of the abstraction generated from the data. For the measure we assume that the two multi-dimensional histograms (original and abstracted) have same number of bins, with bin sizes corresponding to the percentage of data falling into that bin. MHDM is the summation of the difference between the corresponding bins. MHDM ranges from 0.0 to 1.0 with 0 implying the worst case MHDM, and 1 indicating the best case.

One disadvantage of a multi-dimensional histogram is its inability to scale due its high memory requirements [43]. Unfortunately we cannot utilize just 1-dimensional histograms which are less costly, they fail to capture the correlation present in high dimensional data. To overcome the space inefficiency of multi-dimensional histograms [43], we encode the multi-dimensional histogram structure by explicitly associated the multi-dimensional cell address with its cell content value. For example, Figure 3.2 represents the formation of an encoded multi-dimensional histogram. For instance, the cell with dimension 1 at bin 5 and dimesnion 2 at bin 2 and dimension 3 at bin 1 having a value of 6 would be encoded explicitly by the pair ”5*2*1 : 6”.

Building the encoded multi-dimensional histogram: Assume the input tuple with *d* dimensions with data values v_1,v_2,..v_d.

Step I: We partition each of the d dimensions into a number of distinct partitions. For simplicity, we'll assume here that there are exactly n such partitions for each dimension, though other more sophisticated strategies could be employed for bin sizing in the future.

The partitioning of the dimension i is denoted as u_1^i, u_2^i, ... u_n^i with n the number of partitions. For each input tuple v_1,v_2,..v_d, we determine which bin b of dimension i its i^{th} value v_i falls into. Given that each tuple value is mapped to a particular partition, we have *d* partition numbers for a given input tuple. Let us denote this by u_{i1}^1,u_{i2}^2,..u_{id}^d, with i_j the partition number for the dimensions. Thus, the number of 1-dimensional partitions formed directly influence the number of multi-dimensional bins formed.

Step II: We encode the multi-dimensional bin from partition numbers obtained from each dimension by appending the bin numbers into one code, $u_{i1}^1 u_{i2}^2 .. u_{id}^d$ is the multi-dimensional bin corresponding to the example input tuple above. Thus, if most of the d-dimensional cells remain empty, our histogram is relatively small. Most real datasets are very sparse in nature (confirmed by our experimental study in Section 7). Thus this technique saves a lot of memory in practice.

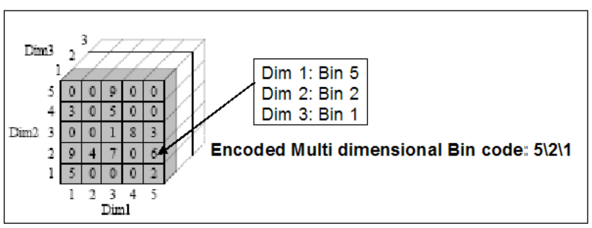

Figure 3.2: Formation of encoded multi-dimensional histogram.

Encoded Multi-dimensional Histogram Properties:

1. *Every encoded multi-dimensional bin is unique*. This implies that every point will fall into only a single bin.

2. Similar data will fall into the same encoded bin.

3. *We do not encode empty bins as bins are formed only when data is to be encoded.* This is a major advantage of our multi-dimensional histogram measure in practice as the real datasets are sparse, we do not waste memory using this approach.

MDHM can be expressed by the following equations:

$$Pb_i = |Po_i - Ps_i| \tag{3.1}$$

where Po_i is the percentage of data that falls into the i-th bin of the original histogram, Ps_i is the percentage of data that fall into the i-th bin of the abstracted histogram, and Pb_i corresponds to their bin difference.

$$Ph = \sum_{i=1}^{N} Pb_i = \sum_{i=1}^{N} |Po_i - Ps_i| \tag{3.2}$$

where Ph is the histogram difference, and N is the number of bins.

$$MHDM = 1.0 - \frac{Ph}{MAX_{Ph}} \tag{3.3}$$

where MAX_{Ph} is the maximum histogram difference.

Noise Elimination: Real world data is often fraught with noise. Noise elimination is very crucial for high quality abstractions. The multi-dimensional histogram of the original data is thus regulated to filter noise. Noise elimination phase consists of eliminating all the bins whose bin count is below a threshold (γ). This threshold γ can either be empirically determined (explained in the experimental section) or set by the analyst. The bin count of a multi-dimensional bin will be below a threshold if:

- The point is a random noise generated by the source.

- The point belongs to the edge of a cluster.

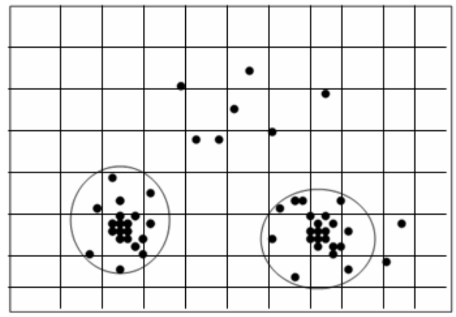

Figure 3.3: Existence of noise in datasets.

Figure 3.3 displays a grid representing a 2-dimensional histogram placed over the data. Ignoring points from low bin counts may have the side effect of ignoring points from the edge of the clusters. However, since we are interested in picking more points from near the center of the cluster rather than its edges, ignoring points from the edges effectively adds more weight to the points in the center. This improves the abstraction quality, as our experimental study confirms (see Section 7). It also decreases the number of multi-dimensional bins to be maintained, increasing the efficiency of the QoS estimator (as further described in Section 3.2.2).

3.2.2 QoS Estimator

The QoS estimator computes the performance-quality trade off by generating a look-up table to indicate the relationship between quality and performance values. This is achieved by computing a relationship between MHDM, sampling level and time required for clustering. Fig 3.4 intuitively represents the look-up table generated by the QoS estimator. The QoS estimator enables the analyst to set the desired quality level in terms of data abstraction quality measure and then to be able to infer time required for clustering. Since sampling is the preliminary step for clustering, the abstraction quality largely depends on the sampling. If the samples chosen for clustering do not represent the original dataset well, the abstraction quality of the clusters can be low. Thus, data abstraction quality is determined by sampling. Various sampling techniques are defined in the literature which can be used in this framework. Palmer et. al devised the strategy of density biased sampling [25]. Density biased sampling is a probability based approach which enables us to sample more from a dense region and less from a sparse region.

MHDM	Sampling level (%)	Time(sec)
.50	0.1	2.38
.60	0.5	4.99
.70	1.2	7.56
.80	2.7	10.16
.90	3	12.33
1.0	3.8	15.3

Figure 3.4: Sample look-up table created by QoS estimator.

According to density biased sampling [25]: Suppose that we have n values x_1, x_2, ... x_n that are partitioned into g groups that have sizes n_1, n_2,, n_g and we want to generate a sample with expected size M in which the probability of point x_i is dependent on size of the group containing x_i.

To bias the sample size, the probability function is defined as:

$$f(n_i) = \frac{\beta}{n_i^e} \tag{3.4}$$

where n_i is the number of points in group i and e is a constant. Number of points selected from group i:

$$n = f(n_i) * n_i \tag{3.5}$$

We define β based on the sample size:

$$E(\text{sample size}) = \sum_{i=1}^{g} E(\text{size of group i}) \tag{3.6}$$

$$M = \sum_{i=1}^{g} n_i f(n_i) = \sum_{i=1}^{g} n_i \frac{\beta}{n_i^e} \tag{3.7}$$

$$\beta = \frac{M}{\sum_{i=1}^{g} n_i^{1-e}} \tag{3.8}$$

Group formation: It is very important to form proper groups for density biased sampling [25] to be effective.The group assignment is done by using the encoded multi-dimensional histogram. The number of bins in the multi-dimensional histogram is equal to the number of groups. Each bin is treated as group of points as defined by density biased sampling.

Algorithm for estimation using density biased sampling:

Input: x= Initial sampling rate

α = Increment in the sampling rate.

Algorithm 1 Populating look-up table

*/*Populating the $lookup\_table$ for performance quality trade off. Initialize by setting M ← x, calculating β from equation 5*/*

while (*MHDM* $\leq$ **1**) **do**
  for each $bin \in$ multi-dimensional histogram **do**
    Number of points selected from each group from Equation 7 ;
  end for
  Compute MHDM for M ;
  Compute time and confidence interval;
  Update *lookup_table with sampling rate and MHDM*;
  M ← *M*+α, compute β ;
  return to 1
end while

Output: lookup table of performance quality trade off.

Look up table is generated after the multi-dimensional histogram for the original data is formed. Starting with sampling level α, number of points falling in each bin are calculated. This enables us to calculate MHDM for sampling level α. It is repeated until MHDM reaches the maximum value of 1.0. The look up table (as shown in Fig.3.4)will have a sampling level, minimum quality level followed for the sampling and the time required for the process to complete. Whenever an analyst chooses a quality value, value closest to it is returned. The granularity of quality-performance trade of is decided by setting small values for α.

3.2.3 Interaction Interface

The interaction module presents the quality estimations of the estimator to the analyst. It allows the analyst to attain information on the quality performance trade off and helps in decision making. The analyst can set one of these three values namely, data abstraction quality (MHDM) value, sampling rate and time for completion of the clustering process. Thus, three cases emerge:

- Analyst sets the desired data abstraction quality and the QoS would indicate implied sampling rate and estimated time for the clustering process to complete.
- If the analyst sets the sampling rate then the best possible quality measure achievable would be indicated along with the expected time for the clustering process to complete.
- If the analyst sets the time for clustering, QoS indicates corresponding sampling rate to be utilized and the achievable data abstraction quality during the allotted time.

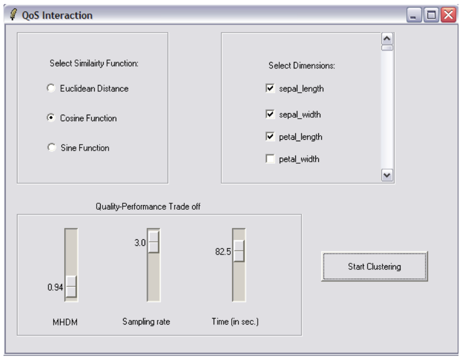

Figure 3.5: QoS sampling interface.

3.2.4 Abstraction Generator

Once the analyst decides the quality and other performance setting he desires, interaction interface passes the sampling level to the abstraction generator. Abstraction generator samples the database using density biased sampling with a sampling level set by the interaction interface. Abstraction generator then passes the generated samples to a clustering algorithm. We can use any existing clustering technique [10, 32, 30] to cluster the data.

Chapter 4

Inclusion of Data Quality

4.1 Total Data Abstraction Quality

Including the quality of the underlying data in the abstraction result is post processing step. MHDM of the data clustered is identical to the value set by the analyst in the pre-processing phase. However, we can also evaluate the performance of the clustering algorithm using a quality measure [14]. One possible clustering quality measure can be average distance of every point from the cluster centers[14]. Once clusters are formed, we can plug these clustering quality measures to find the quality of clustering performed. MHDM can be visualized as a global measure on the entire dataset, where as clustering quality measure gives quality value for each cluster formed.

Every tuple can have quality attributes attached to it. In our earlier work [35], we visualized data quality to have the configuration described in Fig. 2.1. Where, every attribute has quality to it. Record quality value described the quality of the entire tuple, where as dimensional quality describes a quality value for the entire dimension. We only consider the record quality associated with the tuple.

Every cluster consists of data points of the original dataset. Thus, to calculate total abstraction quality, we first calculate data quality of each cluster. To calculate data quality of a cluster, we calculate data quality of all its members using some statistical function.

Many alternative methods are possible, namely

- Arithmetic mean and standard deviation

  Median values

- Geometric mean (assuming geometric distribution)

- Root mean square arithmetic mean

We assume the distribution of data quality values to be uniform. Therefore, we choose to represent the data quality of clusters using the arithmetic mean of the record qualities.

$$CDQ = \frac{\sum_{i=1}^{n} RecordQuality}{n} \tag{4.1}$$

- CDQ: Cluster Data Quality

- Record Quality: Quality of the record.

- n: number of points in the cluster.

Thus, the total data abstraction quality can be calculated as the weighted average of cluster data quality, cluster quality and abstraction quality:

$$TAQ = \frac{\alpha * CQ + \beta * CDQ + \lambda * MHDM}{3} \tag{4.2}$$

- TAQ: Total data abstraction quality

- CQ: Cluster quality

- α: Weight associated with clustering quality

- CDQ: Cluster data quality;

- β: Weight associated with the data quality;

- MHDM is the abstraction quality;

- λ is the weight associated with abstraction quality.

α, β and λ can be user set parameters or can be set to 1 to have arithmetic average.

Chapter 5

Cluster Hierarchy Visualization: InterRing Display

5.1 Cluster Visualization

The purpose of cluster hierarchy visualization is to enable users to visualize, interact and navigate the cluster hierarchy in both data and quality space. As discussed earlier, Radial Space Filling(RSF) display have several advantages over the traditional node link diagrams. Thus, we use InterRing display, which is adaptation of a visualization technique called Sunburst by Stasko and Zhang [39] to visualize the cluster hierarchy in the data space as well as quality space. The InterRing display has the following properties [21]:

- Deeper nodes of the hierarchy are drawn further from the center;
- Child nodes are drawn within the arc subtended by their parents;
- The sweep angle of a leaf node is proportional to the cluster radius;
- The sweep angle of a non leaf node is the aggregation of all its children;
- Color is used to depict the hierarchical structure in the data space and total abstraction quality in the quality space.

- Provision of interactive brushing to analyst so that they can select cluster and view its properties such as total abstraction quality and modify it properties such as name of the cluster.

- Implementation of various other tools such as drill-down/roll-up, zooming in/zooming out operations for InterRing display.

Figure 5.1 shows our cluster hierarchy dialog. Various properties of the InterRing display are depicted in the figure.

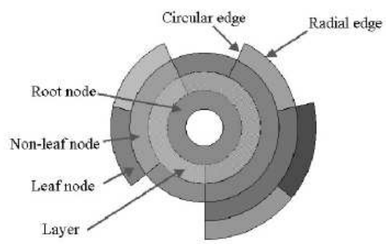

Figure 5.1: InterRing display with labeled components.

5.1.1 Navigation Tools

We provide various navigation tools to allow an analyst to interact and navigate through the cluster hierarchy. These navigation tools can be used to navigate through data as well as quality space.

The navigation tools provided are as follows [21]:

- Distortion:Distortion is a process that results in enlargement selected parts of a display while reducing the screen allocation of other parts. It is helpful in helping the users examine details of the display, and make other interactive operations, such as selection, easier to perform. There are several approaches to distortion to provide focus + context in a radial space filling display [39]. In our InterRing display we provide two types of distortion namely circular and radial distortions. These distortions are easy to use by the users and does not require any extra space for the focus + context display.

– Circular distortion: The basic idea for circular distortion is:

  * A distortion is limited to the angle range of the parent node;
  * A node is increased or decreased in size by decreasing or increasing the size of the siblings;
  * When the sweep angle of a non-leaf node is increased or decreased, all its children's sweep angle is increased or decreased proportionally so that they are always in the angle range of of the parent node.
  * A minimum angle in set for the hierarchy, which is inversely proportional to number of leaf node contained in the hierarchy.

  Circular distortion is helpful for the analyst to select a particular cluster by clicking on a particular edge of the node and analyzing a particular child. Figure 5.2 displays the circular distortion for the InterRing display. In data space, the circular distortion is helpful in studying the child nodes of a particular cluster and in the quality space it helps in understanding the lineage of quality of a particular node and its children.

– Radial distortion: Similar to circular distortion, radial distortion enables an analyst to select a radial edge and distort the display. The maximum radius of the RSF does not changed by distortion. Analyst can pin a radial edge and the thickness of the pinned layer is expanded or contracted by contracting or expanding the thickness of the layers on the dragged edge side. Figure 5.3 displays the radial distortion for analysis of higher nodes in the hierarchy. In

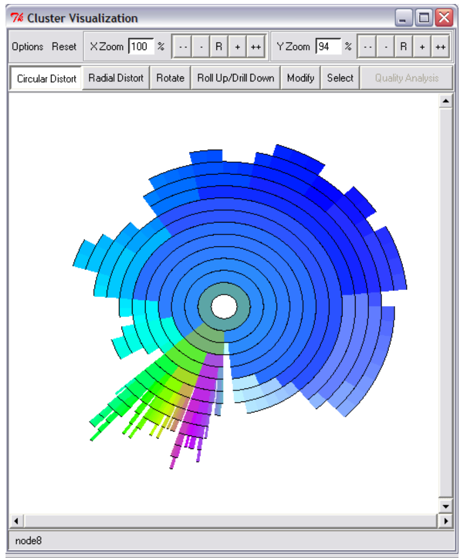

Figure 5.2: Circular distortion in the InterRing display.

data space, it helps in understanding the cluster at a particular level in the tree (studying siblings of a particular node) and in the quality space, helps in comparing quality of siblings.

- Drill-down/Roll-up: Drilling-down/Rolling-up are used to show/hide all the descendants of a cluster. It helps the user to prevent the display of branches that are not of interest for the current analysis. Figure 5.4 displays the drill-down/roll-up operation in the InterRing display. In data space, it helps in concentrating on a sub-tree and in the quality space, it helps in understanding the quality lineage of a particular sub-tree.

- Zooming and Panning: Zooming in/out allows analyst to enlarge the canvas and move around the details of the display.

- Rotation: In rotation mode, InterRing display rotates around the its center in clock wise as well as anti-clockwise direction using a mouse click. This operation helps users rotate clusters of interest to particular angles and avoids cluttering the labels

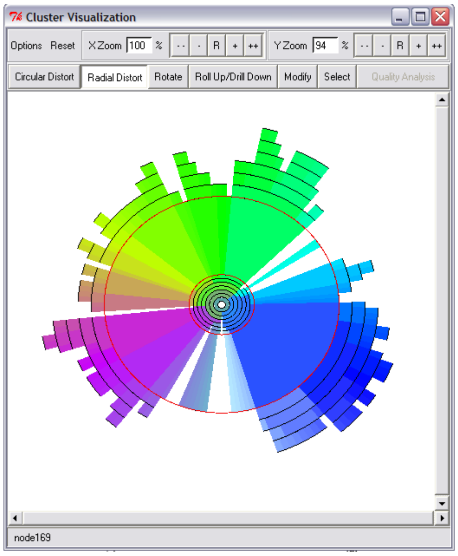

Figure 5.3: Radial distortion in InterRing display.

of the selected clusters. This operation is helpful in orienting the cluster tree to enable the user to study a particular sub-tree.

- Selection: The purpose of selection is to isolate a node or a set of nodes in the hierarchy. These nodes can be used for further quality analysis or renaming the cluster name. Figure 5.5 displays the selection feature of the InterRing display.

- Cluster naming feature: Analyst can select the cluster and rename the cluster to enable them to associate names with the cluster. For example the largest cluster, most important cluster or cluster under current analysis can be named accordingly. Figure 5.6 displays the cluster naming feature of the InterRing display.

5.2 Cluster Quality Visualization

We convey the quality attribute present in the clusters using the InterRing display. There can be various approaches to convey the total abstraction quality of the clusters, for example using the brightness, hue, width(sweep angle) of the clusters or colors. Width of

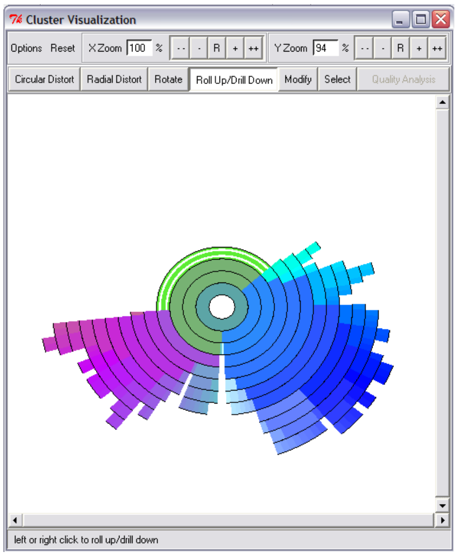

Figure 5.4: Roll up operation in InterRing display.

clusters can change the analytical property of the InterRing. For example, if the analyst wants to know the total abstraction quality of the largest cluster in the hierarchy, mapping the quality according to the width of the cluster might change the visualization. Thus, from brightness, hue and color, we choose color to represent the total abstraction quality of the cluster.

We define a mapping function *f*, which maps a particular quality value to the appropriate color. For example *f* maps perfect quality to green color and worst quality to red color. We use this mapping function to graphically map the cluster using the following function:

G(v, x, f) where:

- *v*: Visual variable used to draw the cluster
- *x*: Numerical value of the total abstraction quality
- *f*: Mapping function for color mapping

The color representation of the quality can be shown in Figure 5.7

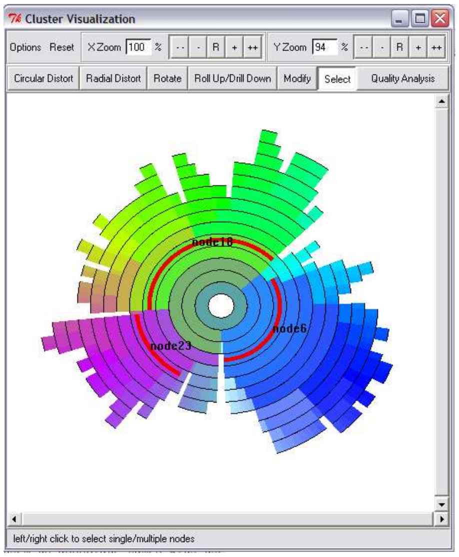

Figure 5.5: Selection feature in Interring Display

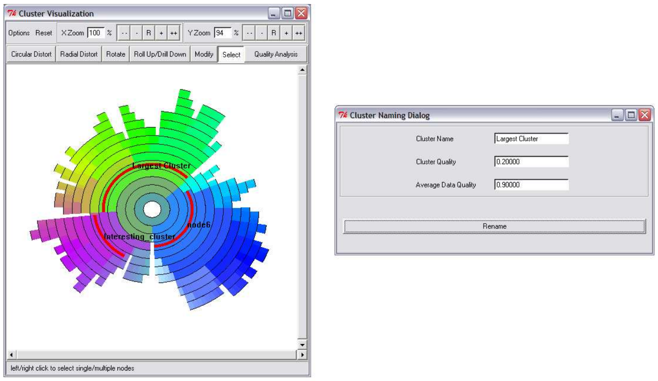

Figure 5.6: Cluster Naming feature in InterRing display.

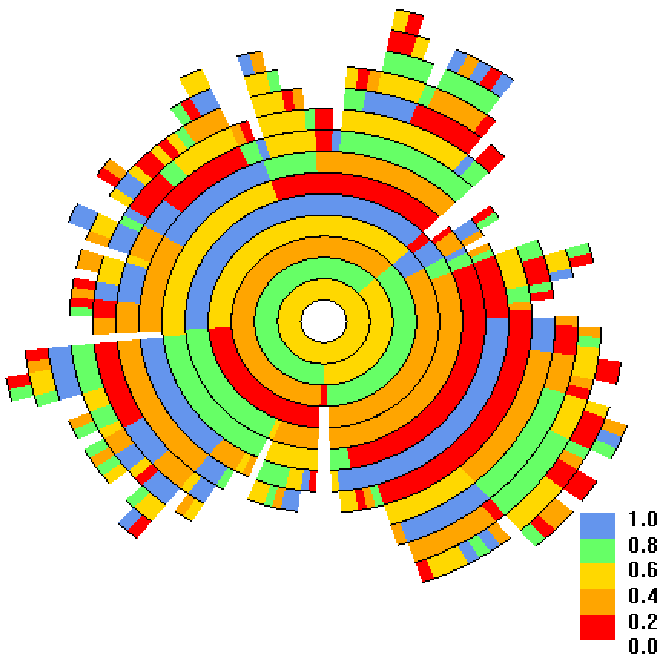

Figure 5.7: Color denoting data quality in InterRing display.

Chapter 6

Implementation

6.1 UML Components

6.1.1 Use Case Diagram:

The interaction of the user with the QoS framework can be depicted using Figure 6.1.

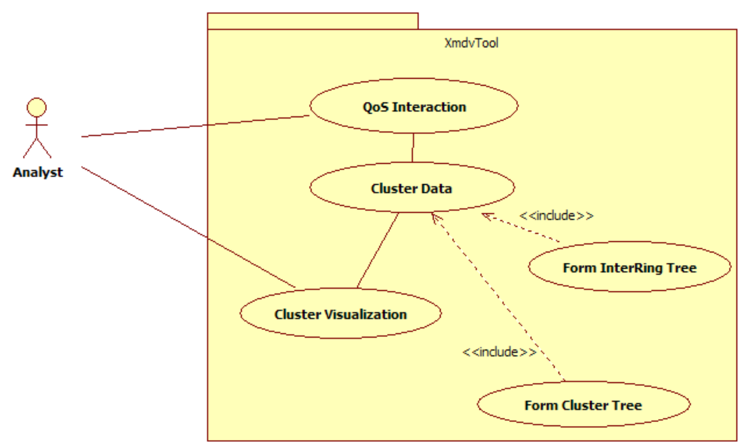

Figure 6.1: Usecase for QoS framework.

User interacts with the QoS franework using the interaction interface. User once satisfied by the time and quality achievable starts the clustering process. QoS then uses the parameters set by the users for clustering the data. Data is clustered in the XmdvTool using adapted Birch algorithm. This algorithm creates clustering tree and InterRing tree.

Cluster tree is used to navigate in the structured space using the structure based brush in heirarchical displays. InterRing tree is used to create the InterRing visualization for data clustering. The user can interact with the clustering result using the InterRing display. Various navigation and interaction tools can be used as discussed in chapter 5.

6.1.2 Class Diagram

The class diagram of the QoS framework along with the clustering process in the Xmdv-Tool can be represented in Figure 6.2.

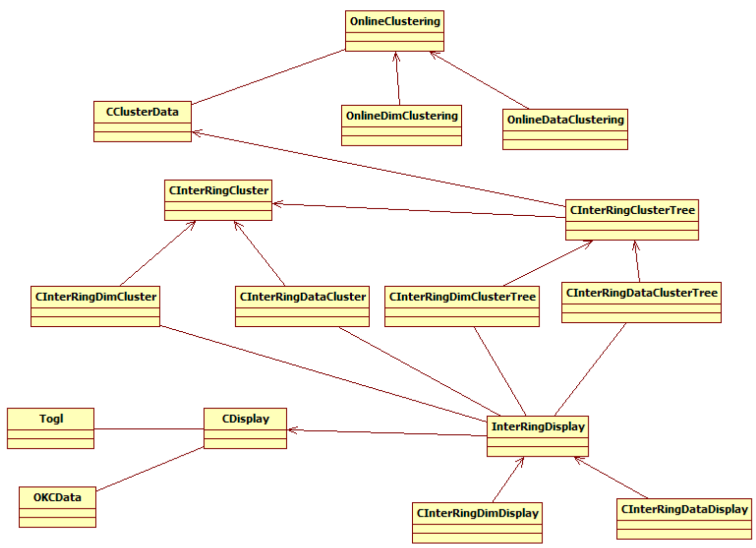

Figure 6.2: Class diagram for QoS framework.

Class OnlineClustering is reponsible for clustering of data and dimensions in Xmd-vTool. This class is the parent class for classes OnlineDimClustering (used for dimensional clustering) and OnlineDataClustering (used for data clustering). Class CCluster-Data is a class that holds the clustering result. CClusterData is associated with CInter-ringClusterTree which is used for displaying the results in the InterRing visualizations. CInterRingClusterTree is associated with CInterRingCluster in one-to-many mapping. CInterRingClusterTree is the parent class for CInterRingDataClusterTree (used for data clustering) and CInterRingDimClusterTree (used for dimensional clustering). Similarly, CInterRingCluster is a parent class for CInterRingDataCluster and CInterDimCluster.

Class CInterRingDisplay is the base class for displaying the InterRing tree. CInterRingDatDisplay and CInterRingDimDisplay are the derived classes used to display the dimensional and data InterRing respectively. Class CDisplay is the base class for displaying the OkcData in Xmdv. CInterRingDisplay derives the CDisplay class. Thus, we make use of multi-level inheritance to cluster the data and display it in the InterRing.

Chapter 7

Experimental Study

We have evaluated the framework using both real and synthetic datasets. The framework is integrated into XmdvTool, a public domain data visualization tool [44] developed at WPI. Experiments were conducted on Pentium 4 (1.66 GHz) running on Microsoft Windows XP with 1.0 GB RAM. We have conducted various experiments for different components of QoS.

7.1 Savings Using Encoding Approach for Multi-dimensional Histogram

In this experiment, we formed encoded multi-dimensional histograms for various real high dimensional datasets such as Iris, Out5d, Cars, Aaup, Census_income and Supercos2 [44]. Figure 7.1 displays the comparisons of the number of bins actually formed and the maximum number of bins possible. As seen from Figure 7.1, savings (difference between maximum possible bins and bins actually formed) increases enormously by using the encoded bin approach. This proves the fact that the real datasets are sparse in nature and our encoding based approach indeed saves memory in practice.

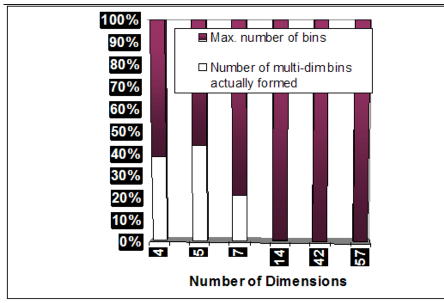

Figure 7.1: Savings for real datasets.

7.2 Effects of Increasing Number of 1-dimensional Partitions

One of the important factors affecting the savings using the encoded approach is the number of 1-dimensional partitions. The savings increase greatly if we form larger number of partitions. However, as the number of 1-dimensional partitions increases the number of multi-dimensional bins increases exponentially. According to Sturges' Rule [23], the number of bins can be based on number of data points. Sturges' Rule is:

$$\text{Number of bins} = 1 + \log_2 \text{Number of data points} \tag{7.1}$$

However, using the sturges rule to form 1-dimensional partitions leads to formation of large number of multi-dimensional bins. Figure 7.2 represents the effect of increasing the number of partitions on number of multi-dimensional bins formed and thus the MHDM. From Figure 7.2, it can be seen that by increasing the number of 1-dimensional partitions the MHDM at a particular sampling rate decreases. Thus, by forming more 1-dimensional partitions better MHDM accuracy can be achieved. However, this adds the overhead of maintaining a large number of bins and effects the performance of QoS estimator.

Therefore, we use an adapted Sturges' Rule and form the bins as:

$$\text{Number of bins} = 1 + \log_2(\text{Number of data points})/3 \tag{7.2}$$

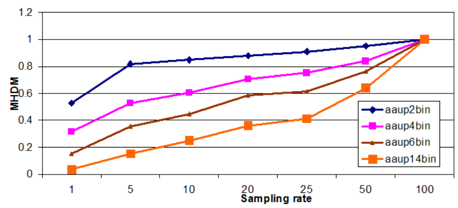

Figure 7.2: Effect of increasing number of 1-d partitions for Aaup dataset.

7.3 Conformance with Cluster Quality Measure

For this experiment, we used synthetic datasets generated with a known number of clusters. We compare the clustering result of the dataset without sampling and after applying QoS. We used a K-means algorithm to find the RMS error [14] between the cluster centers of the original datasets and those of the abstractions generated by QoS. RMS error (ϵ) can be defined as:

$$\epsilon = \sqrt{\frac{\sum_{i=1}^{m}(c_o^i - c_s^i)^2}{m}} \tag{7.3}$$

With c_o^i: original cluster center; c_s^i: cluster center of the abstraction; m: number of clusters.

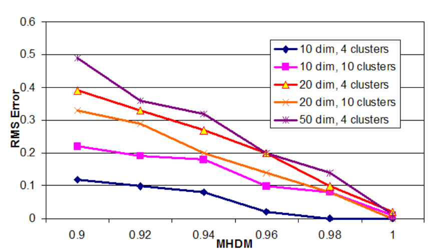

Figure 7.3: Decrease in RMS error with increase in MHDM.

As seen from the Figure 7.3, with the increase in MHDM the accuracy in finding the cluster centers increases and RMS error decreases.

7.4 Need for Noise Elimination

For this experiment, we calculate the average distortion of the dataset. Average distortion is the average distance of a datapoint from its cluster center. Figure 7.4 represents the average distortion with and without noise elimination phase. It can be noticed that in absence of the noise elimination phase, with an increase in MHDM (and thus, the chosen sampling level) the average distortion increases. This is because of the introduction of noise in sampling. Thus, we introduced a noise elimination phase, γ was set at 0.2 percent, eliminating all the bins below the threshold. With the introduction of the noise elimination phase, the average distortion decreases linearly with the increase in MHDM. Since, K-means clustering algorithm was used, the number of clusters formed remains the same. In short, noise elimination stage facilitates formation of dense clusters.

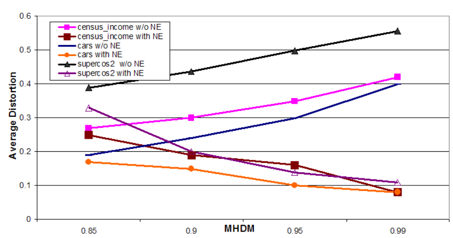

Figure 7.4: Average distortion with and without noise elimination for various real datasets.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Data abstraction quality has implications in many areas such as information visualization, data warehousing and so on. In this thesis, we have proposed a framework to handle quality associated with abstractions at all levels. The contributions of this thesis can be summarized as:

- Multi-dimensional abstraction quality measure to quantify the abstraction quality.
- Density based sampling to empower users to set the abstraction depending on the time/quality requirement.
- Cluster visualization using InterRing display to visualize data and quality.

We have implemented the QoS framework in XmdvTool. Our experiments confirm the merits of our approach.

8.2 Future Work

This work can be extended in the future in many directions, such as

- Using dimensional quality to compute the total abtraction quality.
- Using other visual attributes such as brightness to map the cluster quality.
- Enabling the user to re-cluster the dataset using only a subset of the cluster tree and study the resulting quality of the new cluster tree.
- Allowing the user to select a clustering algorithm and tweak the clustering criteria based on domain knowledge.
- Creating sampling hierarchy (which is cheaper) instead of clustering hierarchy which can be built on the fly and re-sample based on user preferences.

Bibliography

[1] D.F. Andrews. Plots of high dimensional data. *Biometrics, Vol. 28, p. 125-36*, 1972.

[2] J.H. Siegel, E.J. Farrell, R.M. Goldwyn, and H.P. Friedman. The surgical implication of physiologic patterns in myocardial infarction shock. *Surgery Vol. 72, p. 126-41*, 1972.

[3] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association, Vol. 68, p. 361-68*, 1973.

[4] W. Ribarsky, E. Ayers, J. Eble, and S. Mukherjea. Glyphmaker: Creating customized visualization of complex data. *IEEE Computer, Vol. 27(7), p. 57-64*, 1994.

[5] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. *Proc. of Visualization '90, p. 361-78*, 1990.

[6] E.J. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association, Vol. 411(85), p. 664*, 1990.

[7] W.S. Cleveland and M.E. McGill. *Dynamic Graphics for Statistics*. Wadsworth, Inc., 1988.

[8] D.A. Keim, H.P. Kriegel, and M. Ankerst. Recursive pattern: a technique for visualizing very large amounts of data. *Proc. of Visualization '95, p. 279-86*, 1995.

[9] J. LeBlanc, M.O. Ward, and N. Wittels. Exploring n-dimensional databases. *Proc. of Visualization '90, p. 230-7*, 1990.

[10] P. Berkhin. Survey of clustering data mining techniques. In *Technical report, Accrue Software, Inc.*, 2002.

[11] E. Wegman. Hyperdimensional data analysis using parallel coordinates. In *Journal of the American Statistical Association*, pages 664–675, 1990.

[12] Q. Cui, M. Ward, E. Rundensteiner, and J. Yang. Measuring data abstraction quality in multiresolution visualization. In *IEEE Symposium on Information Visualization (InfoVis 2006)*, October 2006.

[13] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: an efficient data clustering method for very large databases. In *ACM SIGMOD*, pages 103–114, 1996.

[14] F. Kovcs, C. Legny, and A. Babos. Cluster validity measurement techniques. In *6th International Symposium of Hungarian Researchers on Computational Intelligence, Budapest*, November 2005.

[15] D.A. Keim and H.P. Kriegel. Visdb: Database exploration using multidimensional visualization. *Computer Graphics & Applications*, 14(5):40–49, 1994.

[16] M. Ankerst, D.A. Keim, and H.P. Driegel. Circle segments: A technique for visually exploring large multidimensional data sets. *Proc. of Visualization '96*, 1996.

[17] P.C. Wong and R.D. Bergeron. Multiresolution multidimensional wavelet brushing. *Proc. of Visualization '96, p. 141-8*, 1996.

[18] D. Keim, M. Hao, J. Ladisch, M. Hsu, and U. Dayal. Pixel bar charts: A new technique for visualizing large multi-attribute data sets without aggregation. *Proc. of Information Visualization 2001, p. 113-120*, 2001.

[19] Y. Fua, M.O. Ward, and E.A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. *Proc. of Visualization '99, p. 43-50*, Oct. 1999.

[20] J. Yang, M. O. Ward, and E. A. Rundensteiner. Interactive hierarchical displays: A general framework for visualization and exploration of large multivariate data sets. In *Computers and Graphics Journal, Vol 27, pp 265-283*, 2002.

[21] J. Yang, M. O. Ward, and E. A. Rundensteiner. An interactive tool for visually navigating and manipulating hierarchical structures. *IEEE Symposium on Information Visualization (InfoVis'02), p. 77-84*, 2002.

[22] Y. Fua, M.O. Ward, and E.A. Rundensteiner. Structure-based brushes: A mechanism for navigating hierarchically organized data and information spaces. *IEEE Visualization and Computer Graphics, Vol. 6, No. 2, p. 150-159*, 2000.

[23] S. Thompson. *Sampling*. John Wiley and Sons, Inc., New York, 2nd Edition, 1992.

[24] F. Olken and D. Rotem. Random sampling from database files: A survey. In *Fifth Int'l Conf. Statistical and Scientific Database Management*, 1990.

[25] C. Palmer and C. Faloutsos. Density biased sampling: An improved method for data mining and clustering. In *ACM SIGMOD*, May 2000.

[26] K. Jain and C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[27] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

[28] R. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. *VLDB'94, p. 144-155*, 1994.

[29] P. Bradley, U. Fayyad, and C. Reina. Scaling clustering algorithms to large databases. *Proc. of KDD '1998, p. 9-15*, 1998.

[30] S. Guha, R. Rastogi, and K. Shim. Cure: an efficient clustering algorithm for large databases. *SIGMOD Record, vol.27(2), p. 73-84*, June 1998.

[31] A. Hinneburg and D. Keim. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. *VLDB'99*, 1999.

[32] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Record, vol.25(2), p. 103-14*, June 1996.

[33] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. *Proceedings of ACM SIGMOD98 International Conference on Management of Data, p. 94-105*, 1998.

[34] K. Chen and L. Liu. Vista: Validating and refining clusters via visualization. In *3rd IEEE International Conf. on Data Mining*, 2003.

[35] Z. Xie, S. Huang, M. Ward, and E. Rundensteiner. Exploratory visualization of multivariate data with variable quality. In *IEEE Symposium on Visual Analytics Science and Technology*, October 2006, pages = 183-190,.

[36] A. Pang. Visualizing uncertainty in geo-spatial data. report for a committee of the computer science and telecommunications board. Technical report, University of California, Santa Cruz, 2001.

[37] J. Stasko, R. Catrambone, M. Guzdial, and K. Mcdonald. An evaluation of space-filling information visualizations for depicting hierarchical structures. *Int. J. Human-Computer Studies, Vol. 53, p. 663-694*, 2000.

[38] T. Barlow and P. Neville. A comparison of 2-d visualization of hierarchies. *Proc. of Information Visualization 2001, p. 131-138*, 2001.

[39] J. Stasko and E. Zhang. Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualization. *Proc. of Information Visualization 2000, p. 57-65*, 2000.

[40] K. Andrews and H. Heidegger. Information slices: Visualising and exploring large hierarchies using cascading, semicircular discs. *IEEE Information Visualization Symposium 1998, Late Breaking Hot Topics Paper, p. 9-12*, 1998.

[41] M. Chuah. Dynamic aggregation with circular visual designs. *Proc. of Information Visualization '98, p. 35-43*, 1998.

[42] J. Widom. A system for integrated management of data, accuracy, and lineage. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR '05), Pacific Grove, California*, January 2005.

[43] H. Wang and K. Sevcik. A multi-dimensional histogram for selectivity estimation and fast approximate query answering. In *conference of the Centre for Advanced Studies on Collaborative research*, pages 328–342, 2003.

[44] Xmdvtool home page. http://davis.wpi.edu/ xmdv/.