CloudVista: Visual Cluster Exploration for Extreme Scale Data in the Cloud

Keke Chen, Huiqi Xu, Fengguang Tian, Shumin Guo

Ohio Center of Excellence in Knowledge Enabled Computing
Department of Computer Science and Engineering
Wright State University, Dayton, OH 45435, USA
{keke.chen, xu.39, tian.6, guo.18}@wright.edu

Abstract. The problem of efficient and high-quality clustering of extreme scale datasets with complex clustering structures continues to be one of the most challenging data analysis problems. An innovate use of data cloud would provide unique opportunity to address this challenge. In this paper, we propose the CloudVista framework to address (1) the problems caused by using sampling in the existing approaches and (2) the problems with the latency caused by cloud-side processing on interactive cluster visualization. The CloudVista framework aims to explore the entire large data stored in the cloud with the help of the data structure *visual frame* and the previously developed VISTA visualization model. The latency of processing large data is addressed by the *RandGen* algorithm that generates a series of related visual frames in the cloud without user's intervention, and a hierarchical exploration model supported by cloud-side subset processing. Experimental study shows this framework is effective and efficient for visually exploring clustering structures for extreme scale datasets stored in the cloud.

1 Introduction

With continued advances in communication network technology and sensing technology, there is an astounding growth in the amount of data produced and made available throughout cyberspace. Cloud computing, the notion of outsourcing hardware and software to Internet service providers through large-scale storage and computing clusters, is emerging as a dominating technology and an economical way to host and analyze massive data sets. Data clouds, consisting of hundreds or thousands of cheap multi-core PCs and disks, are available for rent at low cost (e.g., Amazon EC2 and S3 services). Powered with distributed file systems, e.g., hadoop distributed file system [26], and MapReduce programming model [7], clouds can provide equivalent or better performance than traditional supercomputing environments for data intensive computing.

Meanwhile, with the growth of data volume, large datasets[1] will often be generated, stored, and processed in the cloud. For instance, Facebook stores and processes user activity logs in hadoop clusters [24]; Yahoo! used hadoop clusters to process web documents and generate web graphs. To explore such large datasets, we have to develop novel techniques that utilize the cloud infrastructure and its parallel processing

[1] The concept of "large data" keeps evolving. with existing scales of data, roughly, we consider $< 10^3$ records to be small, $10^3 - 10^6$ to be medium, $10^6 - 10^9$ to be large, and $> 10^9$ to be extreme scale.

power. In this paper we investigate the problem of large-scale data clustering analysis and visualization through innovative use of the cloud.

1.1 Challenges with Clustering Extreme Scale Data

A clustering algorithm tries to partition the records into groups with certain similarity measure [15]. While a dataset can be large in terms of the number of dimensions (dimensionality), the number of records, or both, a "large" web-scale data usually refer to those having multi-millions, or even billions of records. For example, one-day web search clickthrough log for a major commercial web search engine in US can have tens of millions of records. Due to the large volume of data, typical analysis methods are limited to simple statistics based on linear scans. When high-level analysis methods such as clustering are applied, the traditional approaches have to use data reduction methods.

Problems with Sampling.
The three-phase framework, sampling/summarization $\rightarrow$ data analysis on sample data $\rightarrow$ postprocessing/validation is often applied to clustering large data in the single workstation environment (as shown in Figure 1).

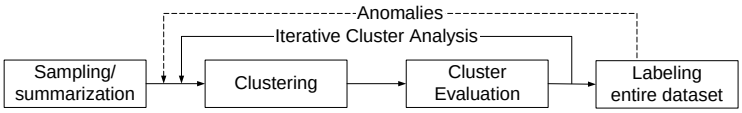

Fig. 1. Three phases for cluster analysis of large datasets

This framework can temporarily address some problems caused by large datasets in *limited scale*. For instance, dealing with complex clustering structures (often the case in many applications) may need clustering algorithms of nonlinear complexity or visual cluster analysis, which cannot be applied to the entire large dataset. With data reduction, the most costly iterative analysis is on the reduced data in the second phase, while we assume the number of iteration involving the three phases is small.

Due to the sampling or summarization phase there is a mismatch between the clustering structure discovered on the sample dataset and that on the entire dataset. To fully preserve the clustering structure, the sampling rate has to be higher than certain lower bound that is determined by the complexity of the clustering structure and the size of the dataset [11]. While the size of entire dataset keeps growing rapidly, the amount of data that the second phase can handle stays limited for a typical workstation, which implies a decreasing sample rate. The previous work in the three-phase visual cluster analysis framework [4] has addressed several problems in extending the clustering structure to the entire dataset under low sample rate, such as missing small clusters, abnormal visual cluster patterns, cluster boundary extension, and unclear secondary clustering structure. These problems become more severe with lower sample rate. Therefore, new processing strategies are needed to replace the three-phase framework for extreme scale datasets.

Problems with Visual Cluster Exploration.
Previous studies have shown that visual cluster exploration can provide unique advan-

tages over automated algorithms [3, 4]. It can help user decide the best number of clusters, identify some irregular clustering structures, incorporate domain knowledge into clustering, and detect errors.

However, visual cluster exploration on the data in the cloud brings extra difficulties. First, the visualization algorithm should be parallelizable. Classical visualization methods such as Principal Component Analysis and projection pursuit [13] involve complex computation, not easy to scale to large data in the parallel processing environment. Second, cloud processing is not optimized for low-latency processing [7], such as interactive visualization. It would be inappropriate to respond to each user's interactive operation with a cloud-based processing procedure, because the user cannot tolerate long waiting time after each mouse click. New visualization and data exploration models should be developed to fit the cloud-based data processing.

1.2 Scope and Contributions

We propose the cloud-based interactive cluster visualization framework, **CloudVista**, to address the aforementioned challenges for explorative cluster analysis in the cloud. The CloudVista framework aims to eliminate the limitation brought by the sampling-based approaches and reduce the impact of latency to the interactivity of visual cluster exploration.

Our approach explores the entire large data in the cloud to address the problems caused by sampling. CloudVista promotes a collaborative framework between the data cloud and the visualization workstation. The large dataset is stored, processed in the cloud and reduced to a key structure "visual frame", the size of which is only subject to the resolution of visualization and much smaller than an extreme scale dataset. Visual frames are generated in batch in the cloud, which are sent to the workstation. The workstation renders visual frames locally and supports interactive visual exploration.

The choice of the visualization model is the key to the success of the proposed framework. In the initial study, we choose our previously developed VISTA visualization model [3] for it has linear complexity and can be easily parallelized. The VISTA model has shown effectiveness in validating clustering structures, incorporating domain knowledge in previous studies [3] and handling moderately large scale data with the three-phase framework [4].

We address the latency problem with an automatic batch frame generation algorithm - the RandGen algorithm. The goal is to efficiently generate a series of meaningful visual frames without the user's intervention. With the initial parameter setting determined by the user, the RandGen algorithm will automatically generate the parameters for the subsequent visual frames, so that these frames are also continuously and smoothly changed. We show that the statistical properties of this algorithm can help identify the clustering structure. In addition to this algorithm, we also support a hierarchical exploration model to further reduce the cost and need of cloud-side processing.

We also implement a prototype system based on Hadoop/MapReduce [26] and the VISTA system [3]. Extensive experiments are conducted to study several aspects of the framework, including the advantages of visualizing entire large datasets, the performance of the cloud-side operations, the cost distribution between the cloud and the application server, and the impact of frame resolution to running time and visualization quality. The preliminary study on the prototype has shown that the CloudVista

framework works effectively in visualizing the clustering structures for extreme scale datasets.

2 CloudVista: the Framework, Data Structure and Algorithms

CloudVista works differently from existing workstation-based visualization. Workstation-based visualization directly processes each record and renders the visualization after the visual parameters are set. In the CloudVista framework, we clearly divide the responsibilities between the cloud, the application server, and the client (Figure 2). The data and compute intensive tasks on large datasets are now finished in the cloud, which will generate the intermediate visual representations - the visual frames (or user selected subsets). The application server manages the visual frame/subset information, issues cloud processing commands, gets the results from the cloud, compresses data for transmission, and delivers data to the client. The client will render the frames, take care of user interaction, and, if the selected subsets are small, work on these small subsets directly with the local visualization system.

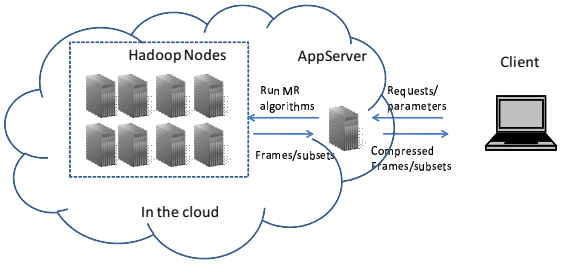

Fig. 2. The CloudVista framework

We describe the framework in three components: the VISTA visualization model, the key data structure "visual frame", and the major data processing and visualization algorithms. We will also include a cost analysis on cloud-side operations at the end of this section.

2.1 The VISTA Visualization Model

The CloudVista framework uses our previously developed VISTA visualization model [3] for it has linear complexity and can be easily parallelized. To make the paper self-contained, we describe the definition of this model and its properties for cluster visualization.

VISTA visualization model is used to map a k-dimensional point to a two dimensional point on the display. Let $\mathbf{s}_i \in \mathbb{R}^2, i = 1, \ldots, k$ be unit vectors arranged in a "star shape" around the origin on the display. $\mathbf{s}_i$ can be represented as $\mathbf{s}_i = (\cos(\theta_i), \sin(\theta_i)), \theta_i \in [0, 2\pi]$, i.e., uniquely defined by θ_i. Let a k-dimensional normalized data point $\mathbf{x} = (x_1, \ldots x_i, \ldots, x_k)$, $x_i \in [-1, 1]$ in the 2D space and $\mathbf{u} = (u_1, u_2)$ be $\mathbf{x}$'s image on the two dimensional display based on the VISTA mapping function.

$\alpha = (\alpha_1, \ldots, \alpha_k), \alpha_i \in [-1, 1]$ are dimensional weights and $c \in \mathbb{R}^+$ (i.e., positive real) is a scaling factor. Formula 1 defines the VISTA model:

$$f(\mathbf{x}, \alpha, \theta, c) = c \sum_{i=1}^{k} \alpha_i x_i \mathbf{s}_i. \tag{1}$$

α_i, θ_i, and c provide the adjustable parameters for this mapping. For simplicity, we leave θ_i to be fixed that equally partitions the circle, i.e., $\theta_i = 2i\pi/k$. Experimental results showed that adjusting α in $[-1, 1]$, combined with the scaling factor c is effective enough for finding satisfactory visualization [3, 4].

Fig. 3. Use a Gaussian mixture to describe the clusters in the dataset.

This model is essentially a simple linear model with dimensional adjustable parameters α_i. The rationale behind the model is

Proposition 1 *If Euclidean distance is used as the similarity measure, an affine mapping does not break clusters but may cause cluster overlapping.*

Proof. Let's model arbitrary shaped clusters with a Gaussian mixture [8]. Let μ be the density center, and Σ be the covariance matrix of the Gaussian cluster. A cluster C_i can be represented with

$$\mathcal{N}_i(\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{k/2}|\Sigma_i|^{1/2}} \exp\{-(\mathbf{x} - \mu_\mathbf{i})' \Sigma^{-1} (\mathbf{x} - \mu_\mathbf{i})/2\}$$

Geometrically, μ describes the position of the cluster and Σ describes the spread of the dense area. After an affine transformation, say $G(\mathbf{x}) = A\mathbf{x} + \mathbf{b}$, the center of the cluster is moved to $A\mu_i + \mathbf{b}$ and the covariance matrix (corresponding to the shape of dense area) is changed to $A\Sigma_i A^T$. And the dense area is modeled with $\mathcal{N}_i(A\mu_i + \mathbf{b}, A\Sigma_i A^T)$. Therefore, affine mapping does not break the dense area, i.e., the cluster. However, due to the changed shapes of the clusters, $A\Sigma_i A^T$, some clusters may overlap each other. As the VISTA model is an affine model, this proposition also applies to the VISTA model.□

Since there is no "broken cluster" in the visualization, any visual gap between the point clouds reflects the real density gaps between the clusters in the original high-dimensional space. The only challenge is to distinguish the distance distortion and cluster overlapping introduced by the mapping. Uniquely different from other models, by tuning α_i values, we can scrutinize the multidimensional dataset visually from

different perspectives, which gives dynamic visual clues for distinguishing the visual overlapping[2].

In addition, since this model is a record-based mapping function, it is naturally parallel and can be implemented with the popular parallel processing models such as MapReduce [7] for large scale cloud-based data processing. Therefore, we use the VISTA model in our framework. Note that our framework does not exclude using any other visualization model if it can efficiently implement the functionalities.

2.2 The Visual Frame Structure

A key structure in CloudVista is the *visual frame* structure. It encodes the visualization and allows the visualization to be generated in parallel in the cloud side. It is also a space-efficient data structure for passing the visualization from the cloud to the client workstation.

Since the visual representation is limited by display size, almost independent of the size of the original dataset, visualizing data is naturally a data reduction process. A rectangle display area for a normal PC display contains a fixed number of pixels, about one thousand by one thousand pixels[3]. Several megabytes will be sufficient to represent the pixel matrix. In contrast, it is normal that a large scale dataset may easily reach terabytes. When we transform the large dataset to a visual representation, a data reduction process happens, where the cloud computation model, e.g., MapReduce, can nicely fit in.

We design the visual representation based on the pixel matrix. The visual data reduction process in our framework is implemented as an aggregation process. Concretely, we use a two dimensional histogram to represent the pixel matrix: each cell is an aggregation bucket representing the corresponding pixel or a number of neighboring pixels (which is defined by the *Resolution*). All points are mapped to the cells and then aggregated. We name such a 2-D bucket structure as "visual frame". A frame can be described as a list of tuples $\langle u_1, u_2, d \rangle$, where (u_1, u_2) is the coordinate of the cell and $d > 0$ records the number of points mapped to the cell. The buckets are often filled sparsely, which makes the actual size of a frame structure is smaller than megabytes. Low resolution frame uses one bucket representing a number of neighboring pixels, which also reduces the size of frame.

Such a visual frame structure is appropriate for density-based cluster visualization, e.g., those based on the VISTA model. The following MapReduce code snippet describes the use of the visual frame based on the VISTA model.

The VISTA visualization model maps the dense areas in the original space to separated or overlapped dense areas on the display. With small datasets, clusters are visualized as dense point clouds, where point-based visualization is sufficient for users to discern clustering structures. With large datasets, all points are crowded together on the display. As a result, point-based visualization does not work. We can use the widely

[2] A well-known problem is that the VISTA model cannot visually separate some manifold structures such nested spherical surfaces, which can be addressed by using spectral clustering [19] as the preprocessing step.

[3] Note that special displays, such as NASA's hyperwall-2, needs special hardware, which are not available for common users, thus do not fit our research scope.

```
1: map(i, x)
2: i: record id, x: k-d record.
3: (u1, u2) ← f(x, α, θ, c);
4: EmitIntermediate((u1, u2), 1)
```

```
1: reduce((u1, u2), v)
2: (u1, u2): coordinate, v: list of counts.
3: d ← 0;
4: for each vi in v do
5:     d ← d + vi;
6: end for
7: Emit(⟨u1, u2, d⟩);
```

adopted heatmap method to visualize the density information - the cells with high density are visualized with warmer colors. With the heatmap method, we can still easily identify clusters from the visualization. We will see some visualization results based on this design in Section 3.

2.3 Algorithms Improving Interactivity

In this section, we describe two major algorithms addressing the latency caused by cloud-side data processing. The first algorithm, RandGen, randomly generates a batch of related frames based on the first frame. The user can then explore the batch of frames locally with the workstation. To further reduce the effect of latency and the need of cloud-side operations, we also develop the algorithms supporting the hierarchical exploration model.

RandGen: Generating Related Frames in Batch Visualization and dimension reduction techniques inevitably bring distance distortion and cause overlapped clusters in lower dimensional space. While it is possible to use algorithms to generate a set of "best" candidate visualization results as projection pursuit [5] does, it is often too costly for large data. Another approach is to allow the user to tune the visual parameters and observe the data in different perspectives to find the possible visual overlapping, which was employed by the VISTA system [3].

In the single workstation mode for medium-size data, the workstation can quickly respond to user's interactive operation and re-generate the visualization by applying the VISTA model to the entire dataset or sample data. However, this interactive model is not realistic if the data processing part is in the cloud. In this section, we develop the RandGen algorithm that can automatically generate a batch of related frames in the cloud based on the parameter setting for the first frame. The collection of frames are passed to the client and the user can spend most time to understand them locally in the workstation. We also prove that the batch of frames generated with RandGen can help users identify the clustering structure.

The RandGen algorithm is a random perturbation process that generates a collection of related frames. Starting from the initial α values that are given by the user, RandGen applies the following small stochastic updates to all dimensional weights simultaneously, which are still limited to the range -1 to +1. Let α_i^{ϕ} represent the α parameter for

dimension i in frame ϕ, the new parameter $\alpha_i^{\phi+1}$ is defined randomly as follows.

$$\delta_i = t \times B,$$
$$\alpha_i^{\phi+1} = \begin{cases} 1 & \text{if } \alpha_i^{\phi} + \delta_i > 1 \\ \alpha_i^{\phi} + \delta & \text{if } \alpha_i^{\phi} + \delta_i \in [-1, 1] \\ -1 & \text{if } \alpha_i^{\phi} + \delta_i < -1, \end{cases} \tag{2}$$

where t is a predefined step length, often set to small, e.g., $0.01 \sim 0.05$, and B is a coin-tossing random variable - with probability 0.5 it returns 1 or -1. δ_i is generated independently at random for each dimension. $\alpha_i^{\phi+1}$ is also bounded by the range [-1,1] to minimize the out-of-bound points (those mapped out of the display). This process repeats until the α parameters for a desired number of frames are generated. Since the adjustment at each step is small, the change between the neighboring frames is small and smooth. As a result, sequentially visualized these frames will create continuously changing visualization. The following analysis shows why the RandGen algorithm can help identify visual cluster overlapping.

Identifying Clustering Patterns with RandGen. We formally analyze why this random perturbation process can help us identify the clustering structure. The change of visualization by adjusting α values can be described by the random movement of each visualized point. Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be the images of the original data record $\mathbf{x}$ for the two neighboring frames, respectively. Then, the point movement is represented as

$$\Delta_{\mathbf{u}} = c \sum_{i=1}^{k} \delta_i x_i \mathbf{s}_i.$$

By definition of B, we have $E[\delta_i] = 0$. Since δ_i are independent of each other, we derive the expectation of $\delta_i \delta_j$

$$E[\delta_i \delta_j] = E[\delta_i] E[\delta_j] = 0, for\ i \neq j.$$

Thus, it follows the expectation of point movement is zero: $E[\Delta_{\mathbf{u}}] = 0$. That means the point will randomly move around the initial position. Let the coordinate $\mathbf{s}_i$ be (s_{i1}, s_{i2}). We can derive the variance of the movement var$(\Delta_{\mathbf{u}}) =$

$$c^2 t^2 \text{var}(B) \begin{pmatrix} \sum_{i=1}^{k} x_i^2 s_{i1}^2 & \sum_{i=1}^{k} x_i^2 s_{i1} s_{i2} \\ \sum_{i=1}^{k} x_i^2 s_{i1} s_{i2} & \sum_{i=1}^{k} x_i^2 s_{i2}^2 \end{pmatrix} \tag{3}$$

There are a number of observations based on the variance. (1) The larger the step length t, the more actively the point moves; (2) As the values s_{ix} and s_{iy} are shared by all points, the points with larger vector length $\sum_{i=1}^{k} x_i^2$ tends to move more actively.

Since we want to identify cluster overlapping by observing point movements, it is more interesting to see how the relative positions change for different points. Let $\mathbf{w}_1$ and $\mathbf{w}_2$ be the images of another original data record $\mathbf{y}$ for the neighboring frames, respectively. With the previous definition of $\mathbf{x}$, the visual squared distance between the pair of points in the initial frame would be

$$\Delta^{(1)}_{\mathbf{w},\mathbf{v}} = ||\mathbf{w}_1 - \mathbf{v}_1||^2 = ||c \sum_{i=1}^{k} \alpha_i (x_i - y_i) \mathbf{s}_i||^2. \tag{4}$$

Then, the change of the squared distance between the two points is

$$\begin{aligned}\Delta_{\mathbf{w},\mathbf{v}} &= 1/c^2(\Delta^{(2)}_{\mathbf{w},\mathbf{v}} - \Delta^{(1)}_{\mathbf{w},\mathbf{v}}) \\ &= (\sum_{i=1} \delta_i(x_i - y_i)s_{i1})^2 + (\sum_{i=1} \delta_i(x_i - y_i)s_{i2})^2 \\ &+ 2(\sum_{i=1} \delta_i(x_i - y_i)s_{i1})(\sum_{i=1} \alpha_i(x_i - y_i)s_{i1}) \\ &+ 2(\sum_{i=1} \delta_i(x_i - y_i)s_{i2})(\sum_{i=1} \alpha_i(x_i - y_i)s_{i2}).\end{aligned}$$

With the independence between δ_i and δ_j for $i \neq j$, $E(\delta_i) = 0$, $\mathbf{s}_{i1}^2 + \mathbf{s}_{i2}^2 = 1$, and $E^2[\delta_i] = t^2\text{var}(B) = 0.25t^2$, it follows the expectation of the distance change is

$$E[\Delta_{\mathbf{w},\mathbf{v}}] = \sum_{i=1}^{k} E^2[\delta_i](x_i - y_i)^2 = 0.25t^2 \sum_{i=1}^{k}(x_i - y_i)^2$$

where , i.e., the average change of distance is proportion to the original distance between the two points. That means, if points are distant in the original space, we will have higher probability to see them distant in the visual frames; if the points are close in the original space, we will more likely observe them move together in the visual frames. This dynamics of random point movement helps us identify possible cluster overlapping in a series of continuously changing visual frames generated with the RandGen method.

Bootstrapping RandGen and Setting the Number of Frames. One may ask how to determine the initial set of α parameters for RandGen. We propose a bootstrapping method based on sampling. In the bootstrapping stage, the cloud is asked to draw a number of samples uniformly at random (μ records, defined by the the user according to the client side's visual computing capacity). The user then locally explores the small subset to determine an interesting visualization, the α parameters of which are sent back for RandGen. Note that this step is used to explore the sketch of the clustering structure. Therefore, the problems with sampling we mentioned in Introduction are not important.

Another question is how many frames are appropriate in a batch for the RandGen algorithm. The goal is to have sufficient number of frames so that one batch is sufficient for finding the important cluster visualization for a selected subset (see the next section for the extended exploration model), but we also do not want to waste computing resources to compute excessive frames. In the initial study, we found this problem is sophisticated because it may involve the proper setting of the step length t, the complexity of the clustering structure, and the selection of the initial frame. In experiments, we will simply use 100 frames per batch. Thorough understanding of this problem would be an important task for our future work.

Supporting Hierarchical Exploration A hierarchical exploration model allows the user to interactively explore the detail of any part of the dataset based on the current visual frame. Such an exploration model can also exponentially reduce the data to be processed and the number of operations to be performed in the cloud side.

We develop algorithms to support such an exploration model. Figure 4 shows the flowchart how the client interacts with the cloud side in this exploration model. Depending on the size of the selected subset of data (ν records), the cloud side may have

different processing strategies. If the selected data is small enough to fit in the client's visualization capacity, i.e., μ records, the cloud will return the subset directly (Case 1). If the rate $\mu/\nu > \xi$, where ξ is an acceptable sampling rate set by the user, e.g., 5%, a uniform sampling is performed on the selected subarea in the cloud to get μ sample records (Case 2). In Case 1 and 2, the subsequent operations on the subset will be handled locally at the client side. Otherwise, if the rate $\mu/\nu < \xi$ that sampling is not an appropriate option, the cloud side will start the RandGen algorithm (Case 3). We will formally analyze the cloud-related cost based on this exploration model in Section 2.4.

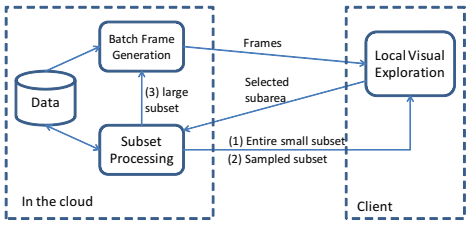

Fig. 4. Interactions between the client and the cloud.

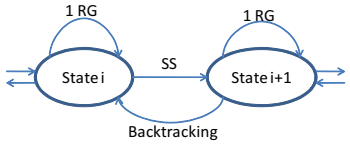

Fig. 5. State transition in terms of operations.

The key operation, subset selection and sampling, should be supported in the cloud. The definition of the selected subset is derived based on the user selected subarea on the current visual frame, and then passed to the cloud together with other visualization parameters. We design a MapReduce algorithm to filter out the selected records based on the area definition. The sampling step can also be appropriately integrated into this step. The details of the algorithms are skipped due to the space limitation.

2.4 A Cost Model for CloudVista

In cloud computing, an important problem is resource provisioning [1]. To understand the interactivity of the system, it is also important to estimate how frequently an exploration will be interrupted for getting results from the cloud. In this section, we will model the exploration process with a Markov chain and derive an estimate to the number of cloud-side operations. The average cost of each operation will be studied in experiments.

The cloud-client interaction can be roughly represented with a Markov chain. Figure 5 shows two sample states of the chain; other states are similarly modeled. The user's interactive exploration can be described as a number of drill-downs on the interested visual areas. Thus, the length of the chain is correlated the number of cloud operations. If the user starts with the state i, she/he may require a RandGen (RG) operation for which the size of data keeps unchanged - let's denote it N_i. Or, she/he can perform a subset selection (SS) to drill down, which moves to the state $i+1$ and the size of dataset is changed to N_{i+1}, correspondingly. This chain extends until the subset can be fully handled locally.

We estimate the length of the chain as follows. Assume a visualization covers n cells, i.e., the aggregation buckets, on the display area on average, and thus the average

density of the cells is N_i/n for state i. We also assume the area the user may select for subsect exploration is about λ percentage of the n cells. So the size of data at state $i+1$ is $N_{i+1} \approx \lambda N_i$. It follows $N_{i+1} = \lambda^{i+1} N_0$. We have defined the client's visualization capacity μ and the acceptable sampling rate ξ. For N_{i+1} records to be handled fully locally by the client, the boundary condition will be $N_i > \mu/\xi$ and $N_{i+1} \leq \mu/\xi$. Plugging $N_{i+1} = \lambda^{i+1} N_0$ into the inequalities, we get

$$\log_\lambda \frac{\mu}{\xi N_0} - 1 \leq i < \log_\lambda \frac{\mu}{\xi N_0},$$

i.e., $i = \lfloor \log_\lambda \frac{\mu}{\xi N_0} \rfloor$. Let the critical value be $\rho = i + 1$. Assume only one RandGen with sufficient number of frames is needed for each state. Since the number of interesting subareas for each level are quite limited, denoted by κ, the total number of cloud operations is $O(\kappa\rho)$.A concrete example may help us better understand the number ρ. Assume the client's visualization capacity is 50,000 records, there are 500 million records in the entire dataset, the acceptable sampling rate is 5%, and each time we select about 20% visual area, i.e., $\lambda = 0.2$, to drill down. We get $\rho = 4$. Therefore, the number of interrupts caused by cloud operations can be quite acceptable for an extreme scale dataset.

3 Experiments

The CloudVista framework addresses the sampling problem with the method of exploring whole dataset, and the latency problem caused by cloud data processing with the RandGen algorithm and the hierarchical exploration model. We conduct a number of experiments to study the unique features of the framework. First, we show the advantages of visualizing the entire large data, compared to the visualization of sample data. Second, we investigate how the resolution of the visual frame may affect the quality of visualization, and whether the RandGen can generate useful frames. Third, we present the performance study on the cloud operations. The client-side visual exploration system (the VISTA system) has been extensively studied in our previous work [3, 4]. Thus, we skip the discussion on the effectiveness of VISTA cluster exploration, although the frame-based exploration will be slightly different.

3.1 Setup

The prototype system is setup in the in-house hadoop cluster. This hadoop cluster has 16 nodes: 15 worker nodes and 1 master node. The master node also serves as the application server. Each node has two quad-core AMD CPUs, 16 GB memory, and two 500GB hard drives. These nodes are connected with a gigabit ethernet switch. Each worker node is configured with eight map slots and six reduce slots, approximately one map slot and one reduce slot per core as recommended in the literature. The client desktop computer can comfortably handle about 50 thousands records within 100 dimensions as we have shown [4].

To evaluate the ability of processing large datasets, we extend two existing large scale datasets to larger scale for experiments. The following data extension method is used to preserve the clustering structure for any extension size. First, we replace the

categorical attributes (for KDD Cup data) with a sequence of integers (starting from 0), and then normalize each dimension[4]. For a randomly selected record from the normalized dataset, we add a random noise (e.g., with normal distribution $N(0, 0.01)$) to each dimensional value to generate a new record and this process repeats for sufficient times to get the desired number of records. In this way the basic clustering structure is preserved in the extended datasets. The two original datasets are (1) **Census 1990 data** with 68 attributes and (2)**KDD Cup 1999 data** with 41 attributes. The KDD Cup data also includes an additional label attribute indicating the class of each record. We denote the extended datasets with Census$_{ext}$ and KDD$_{ext}$ respectively.

3.2 Visualizing the Whole Data

In this experiment, we perform a comparative study: analyzing the visualization results generated with the original VISTA system and the CloudVista framework, on sample datasets and on the entire dataset, respectively. The experiment uses two Census datasets: a sample set of 20,000 records for the VISTA system and an extended dataset of 25 million records (5.3 GB in total) for the CloudVista.

Figure 6 shows the clustering structure with the VISTA system[5]. There are three major clusters - the dense areas in the visualization. This result has been validated with the BestK plot method [4]. Since the Census dataset has been discretized, i.e., all continuous domains are partitioned and discretized, categorical clustering analysis is also applicable. We apply the categorical cluster validation method: BestK plot method to find the best clustering structure [4], which confirms the result visualized in Figure 6. The BestK plot on 1,000 samples shows that the optimal clustering structure has three clusters and a secondary structure has two (these two clustering structures is a part of the hierarchical clustering structure, i.e., two of the three clusters are more similar (closer) to each other than to the third cluster).

Correspondingly, the visualization result in Figure 6 also shows a hierarchical structure based on density: there are three clusters C1, C2.1, and C2.2, while C2.1 and C2.2 are close to each other to form the secondary clustering structure. Except for these major clustering structures, on Figure 6 we have questions about other structural features: (1) Can we confirm that C1 consists of many small clusters? (2) Are there possibly small clusters or outliers between C1 and C2.2? (3) How closely are C2.1 and C2.2 related? These questions are unclear under the visualization of the sample data.

To compare the results, we use the same set of α parameters as the starting point and generate a series of frames with small step length (0.01) on the 25 million records with the CloudVista framework. Figure 7 shows one of these frames. We can answer the above question more confidently with the entire dataset. (1) C1 indeed consists of many small clusters. To further understand the relationship between them, we may need to drill down C1. (2) Small clusters are clearly observed between C1 and C2.2. (3) C2.1 and C2.2 are closed related, but they are still well separated. It is also confirmed that the margin between C1 and C2.x is much larger and clearer than that between C2.1 and C2.2, which is consistent with the secondary structure identified by BKPlot. In addition,

[4] The commonly used methods include max-min normalization or transforming to standard normal distribution.

[5] The dark circles, lines, and annotations are not a part of the visualization (for both Figure 6 and 7). They are manually added to highlight the major observations.

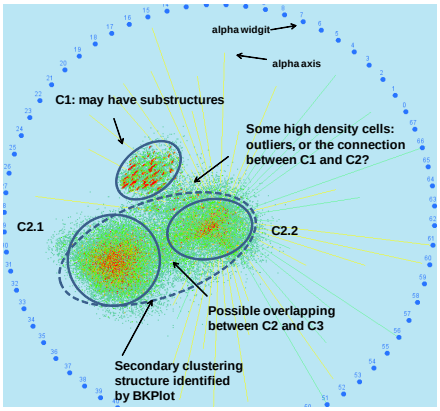

Fig. 6. Visualization and Analysis of Census data with the VISTA system.

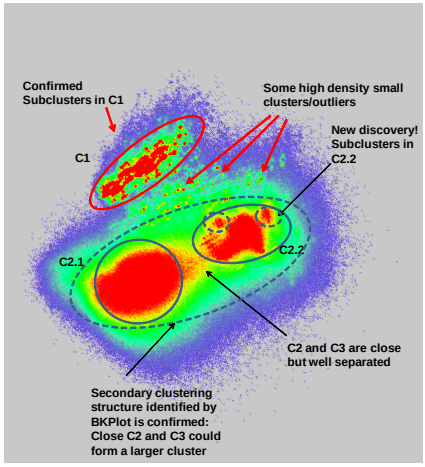

Fig. 7. Visualization and Analysis of 25 Million Census records (in 1000x1000 resolution).

we also find some small sub-clusters inside C2.2, which cannot be observed in Figure 6.

We summarize some of the advantages of visualizing entire large data. First, it can be used to identify the small clusters that are often undetectable with sample dataset; Second, it helps identifying delicate secondary structures that are unclear in sample data. Sample data has its use in determining the major clustering structure.

3.3 Usefulness of Frames Generated by RandGen

We have shown the statistical properties of the RandGen algorithm. In a sufficient number of randomly generated frames by RandGen, the user will find the clustering pattern in the animation created by playing the frames and distinguish potential visual cluster overlaps. We conduct experiments on both the Census$_{ext}$ and KDD$_{ext}$ datasets with the batch size set to 100 frames. Both the random initial frame and the bootstrapping initial frame are used in the experiments. We found in five runs of experiments, with this number of frames, we could always find satisfactory visualization showing the most detailed clustering structure. The video at http://tiny.cc/f6d4g shows how the visualization of Census$_{ext}$ (with 25 millions of records) changes by playing the 100 frames continuously.

3.4 Cost Evaluation on Cloud-Side Data Processing

In this set of experiments, we study the cost of the two major cloud operations: the RandGen algorithm and subset processing. We also analyze the cost distribution between the cloud and the app server.

Lower resolution can significantly reduce the size of the frame data, but it may miss some details. Thus, it represents a potential tradeoff between system performance and

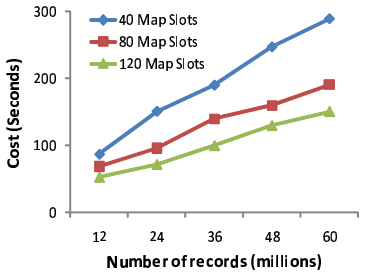

Fig. 8. Running time vs data size for Census$_{ext}$ data (RandGen for 100 frames).

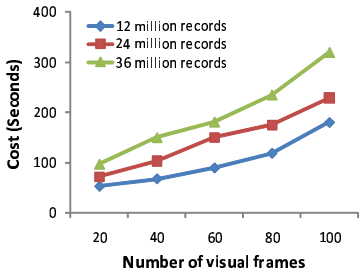

Fig. 9. Running time vs the number of frames for Census$_{ext}$ data.

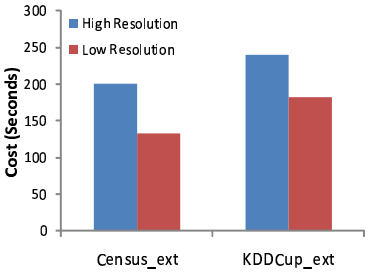

Fig. 10. Cloud processing time vs resolutions for RandGen (100 frames, Census_ext: 25 Million records, KDDCup_ext: 40 Million records).

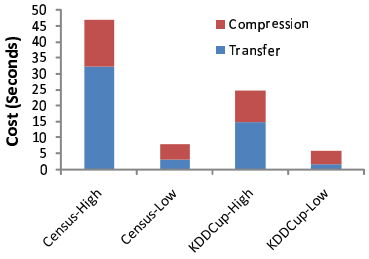

Fig. 11. Cost breakdown (data transfer + compression) in app server processing (100 frames, Census-*: 25 Million records, KDDCup-*: 40 Million records, *-high: 1000x1000 resolution, *-low: 250x250 resolution).

visual quality. Figure 7 in previous discussion is generated with 1000x1000 resolution, i.e., 1 aggregation cell for 1 pixel. Comparing with the result of 250x250 resolution, we find the visual quality is slightly reduced, but the major clustering features are well preserved for the Census$_{ext}$ data. Reducing resolution could be an acceptable method to achieve better system performance. We will also study the impact of resolution to the performance.

RandGen: Figure 8 demonstrates the running time of MapReduce RandGen algorithm with different settings of map slots for the extended census data. We control the number of map slots with Hadoop's fair scheduler. We set 100 reduces corresponding to 100 frames in a batch for all the testing cases[6]. Note that each number in the figures is the average of 5 test runs. The variance is small compared to the average cost and thus ignored in the figures. The running time shows that the MapReduce RandGen algorithm is about linearly scalable in term of data size. With increasing number of map slots, the

[6] We realized this is not an optimal setting, as only 90 reduce slots available in the system, which means 100 reduce processes need to be scheduled in two rounds in the reduce phase.

cost also decreases proportionally. Figure 9 shows the cost also increases about linearly within the range of 100 frames.

We then study the cost distribution at the server side (cloud + application server). The total cost is split into three parts: cloud processing, transferring data to app server from the cloud, and compressing. The following settings are used in this experiment. For RandGen of 100 frames, we compare two extended datasets: 25 million records of Census (Census$_{ext}$) data and 40 million records of KDD Cup (KDD$_{ext}$) data on 15 worker nodes. The results are generated in two resolutions: 1000x1000 (aggregation bucket is 1x1 pixel) and 250x250 (aggregation bucket is 4x4 pixels), respectively. Since the cloud processing cost dominates the total cost, we present the costs in two figures. Figure 10 shows the cost of cloud processing. KDD$_{ext}$ takes more time since its data size is much larger. Also, lower resolution saves a significant amount of time. Figure 11 shows the cost breakdown at the app server, where the suffixes of the x-axis names: "-L" and "-H" mean low and high resolutions, respectively. Interestingly, although KDD$_{ext}$ data takes more time in cloud processing, it actually returns less data in frames, which implies a smaller number of cells are covered by the mapped points. By checking the high-resolution frames, we found there are about 320 thousands of covered cells per frame for census data, while only 143 thousands for KDD cup data, which results in the cost difference in app server processing.

Table 1 summarizes the statistics for different resolutions. We use the amount of data generated by the cloud to represent the communication cost between the cloud and the client (the "compressed data" in Table 1). "Frame size" represents the average number of covered aggregation buckets in each frame; "total time" is the sum of times for cloud processing, transferring from the cloud to the app server, and compressing data. It shows low resolution will have significant cost saving. Low resolution visualization will be very appropriate for exploring higher level clustering structure, where details are less important.

	resolution	frame size	compressed frames	total time(sec)
Census$_{ext}$	High	320K	100MB	247
	Low	25K	9.7MB	141
KDD$_{ext}$	High	143K	45MB	265
	Low	12K	4.6MB	188

Table 1. Summary of the RandGen experiment.

Subset Processing: Subset exploration results in three possible operations: subset RandGen, subset fetching and subset sampling. We have analyzed the number of cloud operations based on the hierarchical exploration model. In this experiment, we let a trained user interactively select interested high-density spots in the frames generated with RandGen and then evaluate how many each of the three operations may be triggered. In each round, 100 frames are generated in each batch with 15 worker nodes on 5.3GB Census$_{ext}$ data or 13.5GB KDD$_{ext}$ data in high resolution. The user browses the frames and randomly selects the high-density subarea to drill down. Totally, 60 drill-down operations are recorded for each dataset.

We summarize the result in Table 2. "Size of Selected Area" represents the average size of the selected area with $\pm$ representing the standard deviation. "Direct" means the number of subsets that will be fully fetched. "Sampling" means the number of sub-

sets that can be sampled. “SS-RG” means the number of subsets, the sizes of which are too large to be sampled - the system will perform a subset RandGen to preserve the structure. “D&S Time” is the average running time (seconds) for each “Direct” or “Sampling” operation in the cloud side processing, excluding the cost of SS-RG, since we have evaluated the cost of RandGen in Table 1.

	Size of Selected Area	# of Cloud Operations			D&S Time(sec)
		Direct	Sampling	SS-RG	
Census$_{ext}$	13896 $\pm$ 17282	4	34	22	36
KDD$_{ext}$	6375$\pm$9646	9	33	18	43

Table 2. Summary of the subsect selection experiment.

Interestingly, the selected areas are normally small: on average about 4% of the entire covered area for both datasets. Most selections, specifically, 63% for Census$_{ext}$ and 70% for KDD$_{ext}$ data, can be handled by “Direct” and “Sampling” and their costs are much less than RandGen.

4 Related Work

Most existing cluster visualization methods cannot scale up to large datasets due to their visual design. Parallel Coordinates [14] uses lines to represent multidimensional points. With large data, the lines are stacked together, cluttering the visual space. Its visual design also does not allow a large number of dimensions to be visualized. Scatter plot matrix and HD-Eye [12] are based on density-plots of pairwise dimensions, which are not convenient for finding the global clustering structure and are not scale to the number of dimensions. Star Coordinates [16] and VISTA [3] models are point-based models and have potential to be extended to handle really large datasets - the work described in the paper is based on the VISTA visualization model. IHD [27] and Hierarchical Clustering Explorer [23] are used to visualize the clustering structures discovered by clustering algorithms, which are different from our purpose of using the visualization system to discover clusters.

Cluster visualization is also a dimensionality reduction problem in the sense that it maps the original data space to the two dimensional visual space. The popularly used dimensionality reduction algorithms such as Principal Component Analysis and Multidimensional Scaling [6] have been applied in visualization. These methods, together with many dimensionality reduction algorithms [21, 22], are often costly - nonlinear to the number of records and thus they are not appropriate for large datasets. FastMap [9] addresses the cost problem for large datasets, but the choice of pivot points in the mapping may affect the quality of the result. Random projection [25] only preserves pairwise distances approximately on average and the precision is subject to the number of projected dimensions - the lower projected dimensions the worse precision. Most importantly, all of these dimensionality reduction methods do not address the common problems - how to detect and understand distance distortion and cluster overlapping. The projection-based methods such as Grand Tour and Projection Pursuit [5] allow the user to interactively explore multiple visualizations to discover possible distance distortion and cluster overlapping, but they are too costly to be used for large datasets. The

family of star coordinates systems [16, 3] address the visual distortion problem with a more efficient way, which is also the basis of our approach. The advantage of stochastic animation in finding patterns, as we do with RandGen, is also explored in graph visualization [2]

The three-phase framework "sampling or summarization – clustering/cluster analysis – disk labeling" is often used to incorporate the algorithms of high time complexity in exploring large datasets. As the size of data grows to very large, the rate between the size of the sampled or summarized dataset to the original size becomes very small, affecting the fidelity of the preserved clustering structure. Some clustering features such as small clusters and the connection between closely related clusters are not easy to be discovered with the sample set [4]. Therefore, there is a need to explore the entire large dataset.

Recently, several data mining algorithms have been developed in the cloud, showing that the hadoop/MapReduce [7] infrastructure is capable to reliably and efficiently handle large-scale data intensive problems. These instances include PLANET [20] for tree ensemble learning, PEGASUS [17] for mining peta-scale graphs, and text mining with MapReduce [18]. There is also an effort on visualizing scientific data (typically, low dimensional) with the support of the cloud [10]. However, none has been reported on visualizing multidimensional extreme scale datasets in the cloud.

5 Conclusion

The existing three-phase framework for cluster analysis on large scale data has reached its limits for extreme scale datasets. The cloud infrastructure provides a unique opportunity to address the problem of scalable data analysis - terabytes or even petabytes of data can be comfortably processed in the cloud. In this paper, we propose the Cloud-Vista framework to utilize the ability of scalable parallel processing power of the cloud, and address the special requirement of low-latency for user-centered visual analysis. We have implemented the prototype system based on the VISTA visualization model and Hadoop/MapReduce. In experiments, we carefully evaluate the unique advantages of the framework for analyzing the entire large dataset and the performance of cloud-side algorithms. The initial results and the prototype system have shown this framework works effectively for exploring large datasets in the cloud. As a part of the future work, we will continue to study the setting of the batch size for RandGen and experiment with larger hadoop cluster.

References

1. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A berkeley view of cloud computing. *Technical Report, University of Berkerley*, 2009.
2. J. Bovey, P. Rodgers, and F. Benoy. Movement as an aid to understanding graphs. In *IEEE Conference on Information Visualization*, pages 472–478. IEEE, 2003.
3. K. Chen and L. Liu. VISTA: Validating and refining clusters via visualization. *Information Visualization*, 3(4):257–270, 2004.
4. K. Chen and L. Liu. iVIBRATE: Interactive visualization based framework for clustering large datasets. *ACM Transactions on Information Systems*, 24(2):245–292, 2006.

5. D. Cook, A. Buja, J. Cabrera, and C. Hurley. Grand tour and projection pursuit. *Journal of Computational and Graphical Statistics*, 23:155–172, 1995.
6. T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman&Hall/CRC, Boca Raton, FL, US, 2001.
7. J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. In *USENIX Symposium on Operating Systems Design and Implementation*, 2004.
8. M. J. (editor). 1998.
9. C. Faloutsos and K.-I. D. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of ACM SIGMOD Conference*, pages 163–174, 1995.
10. K. Grochow, B. Howe, R. Barga, and E. Lazowska. Client + cloud: Seamless architectures for visual data analytics in the ocean sciences. In *Proceedings of International Conference on Scientific and Statistical Database Management (SSDBM)*, 2010.
11. S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large databases. In *Proceedings of ACM SIGMOD Conference*, pages 73–84, 1998.
12. A. Hinneburg, D. A. Keim, and M. Wawryniuk. Visual mining of high-dimensional data. In *IEEE Computer Graphics and Applications*, pages 1–8, 1999.
13. P. J. Huber. Projection pursuit. *Annals of Statistics*, 13(2):435–475, 1985.
14. A. Inselberg. Multidimensional detective. In *IEEE Symposium on Information Visualization*, pages 100–107, 1997.
15. A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31:264–323, 1999.
16. E. Kandogan. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In *Proceedings of ACM SIGKDD Conference*, pages 107–116, 2001.
17. U. Kang, C. E. Tsourakakis, and C. Faloutsos. Pegasus: Mining peta-scale graphs. *Knowledge and Information Systems (KAIS)*, 2010.
18. J. Lin and C. Dyer. *Data-intensive text processing with MapReduce*. Morgan & Claypool Publishers, 2010.
19. A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and algorithm. In *Proceedings Of Neural Information Processing Systems (NIPS)*, 2001.
20. B. Panda, J. S. Herbach, S. Basu, and R. J. Bayardo. Planet: Massively parall learning of tree ensembles with mapreduce. In *Proceedings of Very Large Databases Conference (VLDB)*, 2009.
21. S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
22. L. K. Saul, K. Q. Weinberger, F. Sha, J. Ham, and D. D. Lee. Spectral methods for dimensionality reduction. In *Semi-supervised Learning*. MIT Press, 2006.
23. J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results. *IEEE Computer*, 35(7):80–86, 2002.
24. A. Thusoo, Z. Shao, S. Anthony, D. Borthakur, N. Jain, J. Sen Sarma, R. Murthy, and H. Liu. Data warehousing and analytics infrastructure at facebook. In *Proceedings of ACM SIGMOD Conference*, pages 1013–1020. ACM, 2010.
25. S. S. Vempala. *The Random Projection Method*. American Mathematical Society, 2005.
26. T. White. *Hadoop: The Definitive Guide*. O'Reilly Media, 2009.
27. J. Yang, M. O. Ward, and E. A. Rundensteiner. Interactive hierarchical displays: a general framework for visualization and exploration of large multivariate datasets. *Computers and Graphics Journal*, 27:265–283, 2002.