International Journal of Engineering Research in Electronic and Communication Engineering (IJERECE) Vol 3, Issue 2, February 2016

Signal Processing Techniques For Big Data

[1] Preeti V. Joshi, [2] C.D.Rawat,
[1] PG student,[2] Associate Professor,
Dept. of Electronics and Telecommunication Engineering
VESIT, Chembur, Mumbai,
Maharashtra, India.
[1] preeti.joshi@ves.ac.in,[2] chandansingh.rawat@ves.ac.in

***Abstract*- Big data refers to collection of large and complex data sets which may be in structural, semi-structural, or unstructured format. The processing of such huge amount of data becomes difficult using traditional processing applications. The fundamental challenge is to deal with large volumes of data and extract useful information for future action. As a result, big data requires effective data analysis and processing techniques in order to achieve fast response. This paper explains signal processing techniques for big data.**

Index Terms— Big data, DSP_G, SFT.

I. INTRODUCTION

The world is connecting at a faster rate than ever, which leads to data explosion. Every day, about 2.5 quintillion bytes of data is being generated. Big data is used to increase productivity in business and is extremely valuable for various evolutions in scientific disciplines which creates a lot of opportunities to make great progress.

A big data might be of size petabyte (1024 terabyte) or exabyte (1024 petabyte) containing loosely structured data that is often incomplete or inaccessible.[1] Big data is attributed by 4 V's namely Volume, Variety, Velocity and Veracity. Volume refers to the ability of systems able to handle large amounts of data. This data is represented by heterogeneous and diverse dimensionalities. Heterogeneity is different types of representations for the same observation and diversity involves the features used to represent the same observation. For example, while calculating the literacy rate, the fields for classification may include age, gender, place of living, education acquired etc. on the other hand a survey conducted by health department may involve classification parameters like age, gender, family disease history, income etc. The volume increases as the same individual is being represented by different characteristics depending upon the application[2]. Variety refers to the processing the data of different types such as sparse, uncertain or incomplete data and in various formats like text, graph, audio, video etc. Velocity defines the ability of dealing with regularly or irregularly refreshed data. This also involves collecting of information without relying on centralized control. The fourth V called as veracity is added by IBM scientists indicates that the raw data characteristics must be preserved in processing and synthesizing the data. is to deal with the security and privacy of the data. Sometimes, maintaining privacy prevents disclosing full data set, which often leads to incomplete data fields.[3] To protect privacy, approaches like access control techniques or data anonymization can be implemented. The data sources generate data in real time, so analysis must be performed in real time. Big data processing must also be able to deal with challenges like sharing, storage, visualization.

The reminder of this paper is organized as follows: Section II discusses the techniques that can be used to process big data. Section III emphasizes on real time processing examples for various forms of data. Section IV concludes the paper

II. PROCESSING TECHNIQUES

Big data analysis has been an era of interest in recent times. Due to volume of data involved specialized processing techniques are needed. This section explains processing of big data either by representing it in the form of graph or product graph which is a product of two graphs with smaller dimensions or in sparse domain.

A. Processing using graph:

Discrete signal processing on Graph (DSP_G) extends the methodologies explained in classical signal processing for the data indexed by graph. The graph signal is defined as G = {V, A}, where V = {V_0, V_1,…..V_N} is a set of N nodes corresponding each data elements and A is a weighted adjacency matrix which indicates the relationship among the nodes based on factors like similarity, physical proximity, dependency

etc.[4] Fig (1) shows graph signal of a finite periodic time series having cyclic nature.

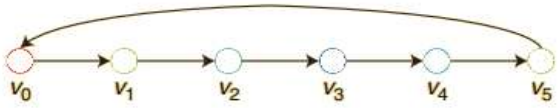

Fig. (1) Graph Signal

The graph signal is then processed by a graph Fourier transform which uses Jordan decomposition of adjacency matrix A. Assuming A to be diagonalizable, the Jordan decomposition is given as:

$$A = V\Lambda V^{-1} \quad (1)$$

Where the columns V_n of matrix $[V_0, V_1, \ldots V_N] \in N\times N$ are the eigenvectors of A and $\Lambda \in N\times N$ is the diagonal matrix of the corresponding eigenvalues. Spectral decomposition of signal s is given by the union of all generalized eigenvectors.

This expansion can be written as:

$$s = V\hat{s} \quad (2)$$

where the vector of expansion coefficients is given by

$$\hat{s} = V^{-1}s \quad (3)$$

The basis of generalized eigenvectors is called as the graph Fourier basis and the expansion given in eq.(3) is called as Graph Fourier transform. The graph Fourier transform matrix is denoted as:[5]

$$F = V^{-1} \quad (4)$$

The inverse graph Fourier transform reconstructs the graph signal from its frequency content by combining graph frequency components weighted by the coefficients of the signal's graph Fourier transform:

$$s = \hat{s}_0 V_0 + \hat{s}_1 V_1 + \ldots\ldots\ldots + \hat{s}_{N-1} V_{N-1} = F^{-1}\hat{s} = V\hat{s} \quad (5)$$

Whenever computation of eigenvectors is not stable, vectors like singular or eigenvectors of the Laplacian matrix can be used as graph Fourier basis.

Frequency analysis is performed to order the graph frequency components according to how much they have changed across the graph i.e. variation of coefficients of frequency components. This is termed as Graph Total Variation. The frequency components with smaller variation are termed as low frequencies while the components with higher variation are high frequencies [6].

B. Processing using product graph:

Product graph is a representation of complex data sets in multilevel and multiparameter ways. The product of two graphs $G_1 = \{V_1, A_1\}$ and $G_2 = \{V_2, A_2\}$ is given as:

$$G = G_1 \Diamond G_2 = \{V, A_\Diamond\} \quad (6)$$

With $V = N_1 . N_2$ nodes and A is the adjacency matrix of order $N_1 N_2 \times N_2 N_1$. Fig.(2) shows a 2D lattice formed by the Cartesian product of two one-dimensional lattices.

There are three commonly defined graph products – Kronecker, Cartesian and Strong, which are used for lowest, highest and mid-spectrum frequency components. The product graph is also processed by graph Fourier transform resulting in reduced number of computations and thereby the computational cost.

The product graphs provides effective computation of Fourier transform on graphs and presents a framework for the development of new methods for analysis of massive data sets.

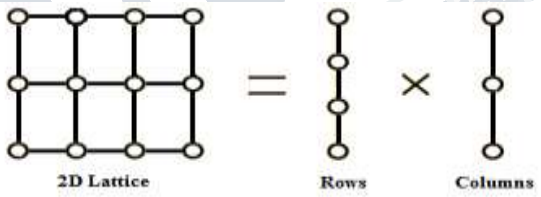

Fig. (2) Product Graph

C. Processing in Sparse domain

The computation of Fourier transform of big data becomes cumbersome with the increase in volume of data. Thus there is need of an algorithm that can compute the Fourier transform in sublinear time using only a subset of input data. This functionality is provided by Sparse Fourier Transform (SFT) which computes the compressed version of DFT in time proportional to number of dominant frequencies. The compressed version contains only a small number of frequencies whose Fourier coefficients have magnitudes greater than the energy of the signal. Since most of the real time signals such as audio, video, GPS signals are sparse in nature, SFT proves be a better alternative for DFT in many big data domains [7].

SFT is computed by using some version of the following three steps:

1. Identification of frequencies whose Fourier coefficients are large in magnitude.
2. Accurate estimation of the Fourier coefficients of the frequencies identified in first step.
3. Subtraction of the contribution of the partial Fourier representation computed by first two steps from the entries of signal before any subsequent repetitions.

The above steps are repeated to calculate the approximate Fourier transform of a signal. Subtracting the estimated coefficients from original ones avoids mistakenly identifying the insignificant frequencies ah being energetic i.e. this stage provides error corrections.

Stage I : Identifying Large Frequencies

This is achieved by

1. Randomly sampling of signal to permute its DFT.
2. Filtering to separate the permuted Fourier coefficients into different frequency bands via filter banks.
3. Estimating energy in subsets of each of frequency bands.

Stage II: Estimating Coefficients

Suppose there are L << N independent and uniformly distributed random samples identified from stage I then the estimate is computed as:

$$\hat{f}'_w = \frac{1}{L}\sum f_{ul} \exp^{-2\Pi i \sigma u l / N} \qquad (7)$$

Stage III: Repetition

It is also called as updation of samples before subsequent repetition. It involves subtraction of Fourier coefficients obtained during stage 2 from the original coefficients. i.e. $f_j - f_l$.

There are several versions of Sparse FFT exist. The computation is mainly divided into two stages: Location loops and estimation loops. Multiple executions of this loops is performed to get the approximate value. Location loops
generate a list of coordinates that have a certain probability of being indices of one of the k significant, nonzero coefficients. Estimation loops are used to determine the frequency coefficients of a given set of Coordinates.

Version 1 uses flat window function to find significant coefficients while version 2 uses specially designed Mansour filter. Version 1 and Version 2 depends on parameters like number of location and estimation loops, number of bins etc. Version 3 is specially designed for exactly K- Sparse signal. It provides some improvements over version 1 and 2. First, estimated coefficients are removed before successive iteration and thus updation of entire signal in not necessary.[8].

III. APPLICATIONS

A. Audio Compression

High compression ratio is desired for the transmission or storage of audio. This can be achieved by exploiting the sparseness of the audio signal. This involves representing sparse signal using linear measurements at a sampling rate much lower than the Nyquist rate. The technique transmits only K-largest frequencies of the signal which saves the transmission bandwidth. The K-largest coefficients of signal of size N are estimated in O (logN√(NKlogN))[9].

The algorithm bins the Fourier coefficients into small number of buckets using a Gaussian function in sub-linear time . As the signal is sparse in frequency domain, each bucket contains only one large coefficient. Once the coefficients are isolated, estimation of these coefficients is performed. The signal in sparse FFT domain can be represented as:

$$\mathbf{X}_s = \mathbf{A}\,\mathbf{X}$$

$$= \begin{bmatrix} 0 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & \cdots & \cdots & 1 & 0 & \cdots \\ \vdots & 0 & \cdots & \cdots & 1 & \cdots \\ \vdots & & 0 & & & 1 \\ & & & \cdots & \cdots & \end{bmatrix} \mathbf{X}$$

One in each row of matrix A corresponds to the non-zero coefficient sampled from sparse vector of the N-point FFT.

Matrix A is adaptive to the signal. As the coefficients are decided beforehand, signal can be optimized to produce the lowest number of coefficients.

B. GPS

The GPS is a navigation system that provides location and time information anywhere where there is an unobstructed line of sight GPS satellites. GPS data is

time-variant, dynamic and large. GPS applications need to be more responsive i.e., the applications should provide real-time information. However, the issue is that these applications need to handle large amount of data from different sources. Hence better computing and storage mechanisms need to be explored to enable such applications [10].

The process of locking on the satellite signal is called as fix which is quite costly and requires millions of hardware multiplications, which leads to high power consumption.

In order to find any location as quickly as possible, an algorithm called QuickSync can be used which uses Sparse Fourier Transform instead of FFT. The main concept used in algorithm is that the position is determined by computing the time required for the signal to travel from each satellite back to the receiver.

Let N be the no. of samples in a signal and $p = \sqrt{(\log N)}$, then the process of finding the location is as follows[11]:

1. Alias $p \times N$ samples into $B = N/p$ buckets.
2. Perform B-point FFT and subsample the output.
3. Subsample the CDMA code of satellite by p and multiply it with result obtained in step 2.
4. Perform IFFT on N/p samples.
5. Find the bucket with maximum magnitude. Check the correlation of each of the p possible time shifts which are aliased into this bucket and pick the shift that gives maximum correlation.

If the signal is too weak, GPS receivers repeat the synchronization algorithm on subsequent signal samples and sum up the output to average out the noise. Squared magnitude of the largest spike is compared to the noise variance in the received signal. If the largest spikes squared magnitude exceeds the noise variance by a desired margin, the algorithm terminates the search and the largest spike will be the correct alignment.

C. Compression of Sensor Measurements

The data collected from weather sensors is required for forecasting the temperature and future weather conditions. For the accurate estimation, measurements are obtained from various sensors installed across the desired location. As the number of observations increases, the signals length increases, which results in complexity in the hardware design and the cost. In order to simplify the processing, these observations can be indexed as a graph nodes and process them using graph Fourier transform.

The testing data set shown in Fig.(3) corresponds to a set of daily temperature measurements collected by 150 weather stations across the United States. Data from each sensor is a separate time series, however, compressing each time series separately requires buffering measurements from multiple days before they can be compressed for storage or transmission. The graph is constructed by connecting each sensor corresponds to node V , $0 \leq n < 150$, to 8 of its nearest neighbors with undirected edges.

The compression is performed in the frequency domain. The Fourier transform of the data set is computed and only C spectrum coefficients with largest magnitudes are retained and others are replaced with zeros. Finally, the inverse graph Fourier transform is performed on the resulting coefficients.

This is a lossy compression scheme, with the compression error given by the norm of the difference between the original data set and the reconstructed one normalized by the norm of the original data set [12].

Fig. (3) Temperature Measurements across the United States reside on the graph that represents the network of weather sensors [4, 5]

The results demonstrate that even for high compression ratios, that is, when the number C of stored coefficients is much smaller than the data set size N= N1.N2, the compression introduces only a small error and leads to insignificant loss of information.

IV. CONCLUSION

In this paper, two signal processing techniques for big data has been discussed. Big data can be processed either by using graphs or in sparse domain. For graphical analysis, big data is represented as a single graph or product of two graphs with lower dimensions. The analysis is then performed by using graph Fourier transform and the frequency contents in the signal are

identified. In sparse domain, the sparse Fourier transform (SFT) computes an approximate or compressed version of Fourier transform using only a subset of the input data, in time smaller than the data set size. This is achieved by implementing algorithms that report only the non-zero or large frequencies and their complex amplitudes rather than a vector containing this information for all frequencies. Since many of the real world signals like audio, video, medical images and GPS signals are sparse, SFT proves to be an efficient technique for processing of big data.

ACKNOWLEDGMENT

I would like to take this opportunity to express my sincere thanks with deep sense of gratitude to my guide Dr. C. D. Rawat, for his valuable support, simulating suggestions and encouragement.

REFERENCES

[1] Trilochan Rout, Mamata Garanayak, Manas Ranjan Senapati, and Sushanta Kumar Kamilla. Big data and its applications: A review. In 2015 International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO), pages 1-5. IEEE, 2015.

[2] Xindong Wu, Xingquan Zhu, Gong-Qing Wu, and Wei Ding. Data mining with big data. IEEE Transactions on Knowledge and Data Engineering, 26(1):97-107, 2014.

[3] Konstantinos Slavakis, Georgios Giannakis, and Gonzalo Mateos. Modeling and optimization for big data analytics:(statistical) learning tools for our era of data deluge. IEEE Signal Processing Magazine, 31(5):18-31, 2014.

[4] Aliaksei Sandryhaila and Jose MF Moura. Big data analysis with signal process-ing on graphs: Representation and processing of massive data sets with irregular structure. IEEE Signal Processing Magazine, 31(5):80-90, 2014.

[5] Aliaksei Sandryhaila and Jose MF Moura. Discrete signal processing on graphs: Graph Fourier transform. In International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6167-6170. IEEE, 2013.

[6] Aliaksei Sandryhaila and Jose MF Moura. Discrete signal processing on graphs:Frequency analysis. IEEE Transactions on Signal Processing, 62(12):3042-3054, 2014.