ISSN:2348-2079

Volume-6 Issue-2

International Journal of Intellectual Advancements and Research in Engineering Computations

Exploring Application Level Semantic for Data Compression

[1]Mrs. C.Navamani M.C.A., M.Phil., M.E,. Assistant Professor,
[2]Mr. M.Jayavel Final MCA
Department of MCA, Nandha Engineering College (Autonomous), Erode-52.
E-Mail ID: navamanimca@gmail.com, jayavel.127@gmail.com

***Abstract*-Current big sensing data processing on Cloud have adopted some data compression techniques. However, due to the high volume and velocity of big sensing data, traditional data compression techniques lack sufficient efficiency and scalability for data processing. Based on specific on-Cloud data compression requirements, propose a novel scalable data compression approach based on calculating similarity among the partitioned data chunks. Instead of compressing basic data units, the compression will be conducted over partitioned data chunks. Map Reduce is used for algorithm implementation to achieve extra scalability on Cloud.**

***Index Terms*- Big Sensing Data, CloudComputing, datachunk, data, compression, similarity model, MapReduce , Scalability.**

I.INTRODUCTION

This Wireless sensor networks have recently come into prominence because they hold the potential to revolutionize many segments of our economy and life, from environmental monitoring and conservation, to manufacturing and business asset management, to automation in the transportation and health care industries. The design, implementation, and operation of a sensor network requires the confluence of many disciplines, including signal processing, networking and protocols, embedded systems, information management and distributed algorithms. Such networks are often deployed in resource-constrained environments, for instance with battery operated nodes running un-tethered.

In this paper, study the application of CS with random walks for data gathering in WSNs. We adopt the standard random walk algorithm to collect random measurements along multiple random paths. However, such an approach will lead to the non-uniform selection of measurements, which is different from uniform sampling in the traditional CS theory. It is still unknown whether such an approach can be used to recover sparse signals in a WSN scenario. To the best of our knowledge, the problem of data gathering with CS based on random walk algorithm has not been significantly investigated.

However, they focus on designing rate less code to exploit the correlation of the signals and use belief propagation decoding algorithm. Analysis the problem of compressive sensing sparse signals over the graph for network tomography application. In their work, the authors model the network as a random graph (G n, p) in which an edge exists between any two nodes with probability p in the network with n nodes. However, such a model is not appropriate to wireless sensor networks since connectivity in such networks is dependent on the distance between two nodes. Moreover, their work focuses on the case where the mixing time has an upper bound as the number of nodes n grows, which is different from our case.

- Motivated by data gathering applications, we propose a simple measurement collection algorithm with random walks for compressive sensing.
- We provide mathematical foundations to use such an approach for understanding the intrinsic principles between CS theory and graph theory.
- We also demonstrate the effectiveness of the proposed scheme through simulations. The simulation results show that the proposed scheme can significantly reduce the communication cost for data gathering applications

Objectives

- To propose a novel compression algorithm to compress the location data of a group of moving objects with or without loss of information.

- To avoid transmitting unnecessary and redundant data by transmitting only the local grouping results to a base station (the sink), instead of all of the moving objects' location data.
- To use two phase - 2D algorithm, which utilizes the discovered group movement patterns shared by the transmitting node and the receiving node to compress data.
- To effectively reduce the amount of delivered data and enhances compressibility.

II.LITERATURE SURVEY

Distributed Sparse Random Projections for Refinable Approximation Wei Wang, Minos Garofalakis, Kannan RamchandranConsider a large-scale wireless sensor network measuring compressible data, where n distributed data values can be well-approximated using only k << n coefficients of some known transform. They address the problem of recovering an approximation of the n data values by querying any L sensors, so that the reconstruction error is comparable to the optimal k-term approximation. To solve this problem, present a novel distributed algorithm based on sparse random projections, which requires no global coordination or knowledge.

The key idea is that the sparsity of the random projections greatly reduces the communication cost of pre-processing the data. The algorithm proposed in this study [1] allows the collector to choose the number of sensors to query according to the desired approximation error. The reconstruction quality depends only on the number of sensors queried, enabling robust refinable approximation. Suppose a wireless sensor network measures data which is compressible in an appropriate transform domain[2,3], so that n data values can be well-approximated using only k << n transform coefficients.

In order to reduce power consumption and query latency, want to pre-process the data in the network so that only k values need to be collected to recover the data with an acceptable approximation error. Computing the deterministic transform in a distributed manner is difficult in an unreliable wireless network, requiring global coordination and knowledge of the network state. In addition, one must then locate the largest transform coefficients in the network to recover the best approximation.

There is a rich literature on using random projections to approximate functions of data. It is a well-known result from compressed sensing [4,5] that O(poly(k, log n)) random projections of the data are sufficient to recover a representation very close to the optimal approximation using k transform coefficients. In this study, show that O(poly(k, log n)) sparse random projections are sufficient to recover a data approximation which is comparable to the optimal k-term approximation, with high probability. The expected degree of sparsity, that is, the average number of nonzeros in each random projection vector, can be O(log n). In fact, there is a trade-off between the sparsity of the random projections and the number of random projections needed.

Second, present a distributed algorithm, based on sparse random projections, which guarantees recovery of a nearoptimal approximation by querying any O(poly(k, log n)) sensors. Their algorithm effectively acts as an erasure code over real numbers, generating n sparse random projection coefficients out of which any subset of O(poly(k, log n)) is sufficient to decode. Since sparsity of the random projections determines the amount of communication, the communication cost can be reduced to O(log n) packets per sensor, routed to randomly selected nodes in the network. There is a corresponding trade-off between the pre-processing communication cost and the number of sensors that need to be queried to recover an approximation with acceptable error.

III.SYSTEM ANALYSIS

Existing System

In the existing system, data size reduction, reduce the processing time cost and release the iterations in processing big sensing data, a novel technique based on data chunk partitioning is implemented for effectively processing big data, especially streaming big sensing data on Cloud. With this novel technique, big sensing data stream will be filtered to form standard data chunks at first based on our predefined similarity model. Then, the coming sensing data stream will be compressed according to the generated standard data chunks. With the above data compression, to improve the data compression efficiency by avoiding traditional compression based on each data unit, which is space and time costly due to low level data traverse and manipulation.

At the same time, because the compression happens at a higher data chunk level, it reduces the chance for introducing too much usage of iteration and recursion which prove to be main trouble in processing big graph data. The existing chunk based compression; data sets should be compression block by block. For big graph data and lots of network data, the topology and structure information has big influence for data processing and it should not be ignored.

Drawbacks:

The existing system has following disadvantages,

- Sequential patterns consider the characteristics of all objects.

- Sequential patterns lack information about a frequent pattern's significance regarding individual trajectories
- Sequential patterns carry no time information between consecutive items.
- Make them unsuitable for location prediction and similarity comparison.

Proposed System

Discovering the group movement patterns is more difficult than finding the patterns of a single object or all objects, because it is required to jointly identify a group of objects and discover their aggregated group movement patterns. The constrained resource of WSNs should also be considered in approaching the moving object clustering problem.

A clustering algorithm is proposed to find the group relationships for query and data aggregation efficiency. First, since the clustering algorithm itself is a centralized algorithm, the project further considers systematically combining multiple local clustering results into a consensus to improve the clustering quality and for use in the update-based tracking network. Second, when a delay is tolerant in the tracking application, a new data management approach is required to offer transmission efficiency, which also motivates this study. The project defines the problem of compressing the location data of a group of moving objects as the group data compression problem.

The thesis formulates a moving object clustering problem that jointly identifies a group of objects and discovers their movement patterns. The application-level semantics are useful for various applications, such as data storage and transmission, task scheduling, and network construction. The thesis proposes an efficient distributed mining algorithm to minimize the number of groups such that members in each of the discovered groups are highly related by their movement patterns.

Specifically, the 2P2D algorithm comprises a sequence merge and an entropy reduction phases. In the sequence merge phase, a Dynamic Merge Compressive algorithm is proposed to merge and compress the location data of a group of objects.

ADVANTAGES OF PROPOSED SYSTEM

- The thesis proposes a novel compression algorithm to compress the location data of a group of moving objects with or without loss of information.
- It avoids transmitting unnecessary and redundant data by transmitting only the local grouping results to a base station (the sink), instead of all of the moving objects' location data.
- Using two phase - 2D algorithm, the proposed system utilizes the discovered group movement patterns shared by the transmitting node and the receiving node to compress data.
- The proposed compression algorithm effectively reduces the amount of delivered data enhances compressibility.

IV.METHODOLOGY

DATA CHUNK SIMILARITY

The similarity is given by distance between objects in this geometric space; the closer together two objects are, the more similar they are. Normally, the similarity is described with a cosθ between two vectors and fraction between two matrix norms ||x|| and ||y||. the numerical data similarity of θ is defined and denoted as $Sim_{n1}(x,y)$.

The above similarity computation can be categorized as a typical geometry data similarity finding process. It is designed from the common cosine similarity model. The cosine similarity between two vectors (or two documents on the Vector Space) is a measure that calculates the cosine of the angle θ between them. To measure similarity between two vectors x and y, a popular similarity function is the inner product including the cosine similarity and it can measure the big data chunk similarity more accurately under our big sensing data feature assumption.

SIMILARITY OF TEXT DATA

For string type and text type similarity, a dual variable length hidden Markov model is used and updated in this work for calculating similarity between text data. Suppose there are a string pair p(str1, str2), and a time stamp series t= t1, t2, …, tn. We can define the joint probability PR of each pair by the state time stamp series. Generally there are multiple state transitions that produce a given pair of strings. If the set of state sequences that produces a pair p is denoted as τ (p). Then the string similarity of the pair p is defined as the maximum alignment probability.

In data chunk based compression, data sets should be compression block by block. For big graph data and lots of network data, the topology and structure information has big influence for data processing and it should not be ignored.

FORMATION OF DATA CHUNKS

Due to the huge volume of the big sensing data, the only data size reduction at that level is not enough. Novel data compression techniques should be developed to dramatically reduce the stored data size and time cost for data manipulation. Instead of compressing data unit one by one at sampling stage and traditional compression techniques which compare data units at a low level, some compression based on huge

data blocks in big sensing data should be developed. Considering an example, all the collected sensing data will be transmitted back to node 0. With a traditional data storage strategy, the data collected from sensing node 1 to sensing node 12 will be stored one by one with a certain predefined order. Under this storage theme, the traditional compression happens at each sensing node level.

The data will not be compressed by encoding or data prediction one by one. It is similar to high frequent element compression. The difference is that the frequent element compression recognizes only simple data units; whereas our data chunk based compression recognizes complex data partitions and patterns during the compression process. Similar to chess games, variations and patterns are well studied and predefined, and most of operations will happen at variation level. For example, a data unit in a big data set S is denoted as Xi, and the first data unit should be denoted as X1. In other words, S is a stream data series, denoted as S={X1, X2… Xm}, where m is the number of data units streaming in. If the following data unit vectors X2 and X1, can be used to calculated $Sim_{n1}(\overrightarrow{X1}, \overrightarrow{X2})$ and $Sim_{n2}(\overrightarrow{X1}, \overrightarrow{X2})$. At the same time, if it is text type data, the dual variable length hidden Markov model will be used to calculate similarity.

DATA CHUNK BASED COMPRESSION

To compress the big data set S from vector uj, there is no need for the compression algorithm to navigate uj one by one. Whereas, the standard chunks stored in S' will be used to compress the in-coming vectors series uj chunks by chunks. For example, with the generated standard chunks set S', a whole block of data uj to uj+r will be compared to vr in S' firstly. If the distance between vr and uj+r, Dis(vr , {uj,…, uj+r})>T can be calculated, the uj+r will be recursively decomposed with the sequence of subsets from {uj, uj+1, uj+2, …, uj+r}.

SIMILARITY BASED COMPRESSION

To implement the data compression on Cloud, two important stages, standard chunk generation and chunk based compression are essential. At the first stage, the standard data chunks are generated. The algorithm for selecting those chunks can be per formed before the real data compression by centralized computer systems. So, a centralized algorithm is offered for describing the whole process of standard data chunk generation. At the second stage of big data compression, the storage and time saving is mainly achieved by chunk based compression and scalability of Cloud. The chunk based compression is introduced by the algorithm itself, and the scalability is introduced by designing the compression algorithm with MapReduce. In other words, the compression algorithms conclude two parts, "Mapper" side algorithm and "Reducer" side algorithm.

The Mapper side algorithm takes the S and S' as its input. The output of Mapper side algorithm is a data set S which its data element tagged. All the tagged elements are able to be compressed and decompressed based on S'. The total number of elements in the standard data chunks set, S' is calculated and stored in L. Because during the compression process, the data vectors from S is selected chunk by chunk with the length L, the algorithm needs to record the starting point of the data element. The recursive similarity comparison function is called again to tag any data element in S to find any xi∈S which could be compressed. The redcue function takes tagged big data set, S as its input. any tagged data element in S is compressed by the compress function of any element in S. After each function call of compress, the storage should be updated by a function of update.

COMPRESSION WITH GROUP MOVEMENT PATTERNS

Data compression can reduce the storage and energy consumption for resource-constrained applications. To reduce the amount of delivered data, propose the 2P2D algorithm which leverages the group movement patterns to compress the location sequences of moving objects elaborately. The approach includes the sequence merge phase and the entropy reduction phase to compress location sequences vertically and horizontally. In the sequence merge phase, propose the Merge algorithm to compress the location sequences of a group of moving objects. Since objects with similar movement patterns are identified as a group, their location sequences are similar.

The Merge algorithm avoids redundant sending of their locations, and thus, reduces the overall sequence length. It combines the sequences of a group of moving objects by 1) trimming multiple identical symbols at the same time interval into a single symbol or 2) choosing a qualified symbol to represent them when a tolerance of loss of accuracy is specified by the application. The technique trims and prunes more items when the group size is larger and the group relationships are more distinct.

In the entropy reduction phase, propose the Replace method that utilizes the group movement patterns as the prediction model to further compress the merged sequence. The Replace algorithm guarantees the reduction of a sequence's entropy, and consequently, improves compressibility without loss of information.

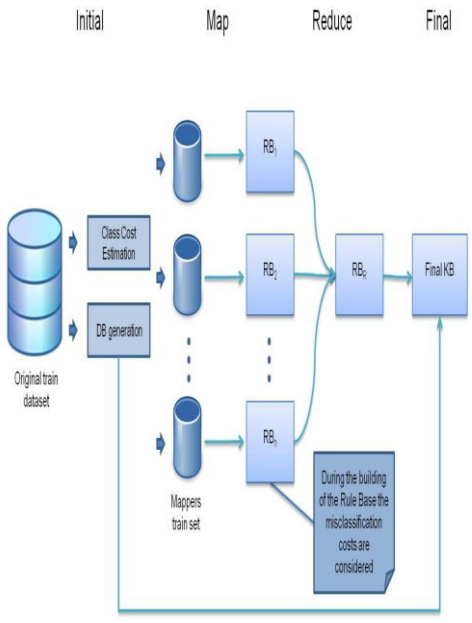
SEQUENCE MINE ALGORITHM

The GMPMine algorithm extracts the movement patterns from the location sequences by learning a PST for each object. In this module, a new similarity measure sim_p to compare the similarity of two objects is proposed. For each of their significant movement patterns, the new similarity measure considers not merely two probability distributions but also two weight factors, i.e., the significance of the pattern regarding to each PST. The similarity score simp of oi and oj based on their respective PSTs, Ti and Tj. The similarity score is equally divided into a threshold value so that the objects can be grouped in a single cluster if the similarity score falls within that range.

COMPRESSION ALGORITHM WITH GROUP MOVEMENT PATTERN

To reduce the amount of delivered data, the 2P2D algorithm is proposed which leverages the group movement patterns derived to compress the location sequences of moving objects elaborately. The algorithm includes the sequence merge phase and the entropy reduction phase to compress location sequences vertically and horizontally. In the sequence merge phase, the Merge algorithm is proposed to compress the location sequences of a group of moving objects.

ARCHITECTURAL DESIGN



V.CONCLUSION

In this study, proposed a scalable data compression based on similarity calculation among the partitioned data chunks with Cloud computing. A similarity model was developed to generate the standard data chunks for compressing big data sets. Instead of compression over basic data units, the compression was conducted over partitioned data chunks. The MapReduce programming model was adopted for the algorithms implementation to achieve some extra scalability on Cloud. In addition to that propose a distributed mining algorithm, which consists of a local GMPMine algorithm. With the discovered information, the 2P2D algorithm is devised, which comprises a sequence merge phase and an entropy reduction phase. In the sequence merge phase, we propose the Merge algorithm to merge the location sequences of a group of moving objects with the goal of reducing the overall sequence length.

REFERENCES

[1] S. Tsuchiya, Y. Sakamoto, Y. Tsuchimoto and V. Lee, "Big Data Processing in Cloud Environments," FUJITSU Science and Technology Journal, 48(2): 159-168, 2012.

[2] "Big data: science in the petabyte era: Community cleverness Required" Nature 455 (7209): 1, 2008.

[3] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "A view of cloud computing," Communications of the ACM 53(4): 50-58, 2010.

[4] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems 25(6): 599-616, 2009.

[5] L. Wang, J. Zhan, W. Shi and Y. Liang, "In cloud, can scientific communities benefit from the economies of scale?" IEEE Transactions on Parallel and Distributed Systems 23(2): 296-303, 2012