Proficient way of Managing Multidimensional Aggregate Data in Unstructured Peer to Peer Networks

Srinivas Raj
LIET, Hyderabad,
India

Murtuza Ahmed Khan
LIET, Hyderabad,
India,

Abstract: the networks which are Peer-To-Peer (P2P) have become very trendy in the last few years. Currently, they are the most common approach to exchange data among large societies of users in the file sharing situation. The capable way of managing storage and retrieval of multidimensional data is achieved by proposed structure which ensures vigorous query development. This structure is based on P2P network, where huge collection of data to be stored. This data is divided into subparts and built up an index on set of each compressed data and this data is to be distributed across p2p networks. This compressed data supports efficient data extraction of information. A duplication method provides suitable coverage of index and metadata by considering network conditions and workload of the queries.

Keywords- multidimensional data, indexing, compression, p2p network, synopsis

1. INTRODUCTION

Currently, they are the most general approach for exchanging data among large societies of users in the file sharing environment. In order to make contributors really independent, they should be forced no restriction on storage and computational resources to be shared, as well as on the consistency of their network connection **[1]**. These necessities make conventional distributed frameworks unsuitable and suggest the implementation of a solution based on an unstructured P2P network, where peers are neither responsible of coordination tasks. The aim is inventing a P2P-based framework supporting the analysis of multidimensional past data. In particular, the efforts are devoted to combine the facilities of P2P networks and data compression to provide a support for the assessment of range queries, possibly trading off effectiveness with accuracy of answers **[2]**. The structure should enable members of an association to cooperate by sharing their resources to host data and perform aggregate queries on them, while preserving their independence. The management of compressed data on unstructured P2P Networks is an interesting issue, but poses several research challenges, which are discussed in the following.

1.1 Compression

A compression technique must be developed which is able to create distributed data supporting the efficient assessment of aggregates, possibly affected by bearable error rates **[3]**. However, in this case, although the rate of disk storage is continuously and quickly decreasing, it may still be hard to find peers for which hosting replicas has a minor cost, while independence is a requirement in the present setting. Using conventional compression techniques, synopsis provides reasonable error rates may have a non-negligible size. Although compressing the data certainly makes replication less resource consuming, replicating the entire synopsis each time would require storage and network resources that could be saved if only some specific portion of the synopsis could be replicated **[4]**. The replication is mandatory in the P2P setting; both to contrast the volatility of peers and to prevent peers from being overloaded are recalled. These drawbacks would be overcome if the compressed synopsis were subdivided into tiny sub synopsis which are independently replicated and distributed on the network when needed. Peers would, therefore, be asked to host replicas of small chunks of data. This way, the independence requirement would not result in a limit on the overall size of the synopsis.

1.2 Indexing

The best way to address this issue is to design an indexing method that supports the efficient location of the sub synopsis involved in the query valuation. In the literature, there are several works proposing distributed indexing methods, where indexes are variants of R-Trees which are partitioned and distributed among the nodes of the network. According to these approaches, nodes of the networks are allocated groups of nodes of the R-tree, and maintain references to hosts which are assigned other nodes of the R-tree. The connection between hosts and R- tree nodes is fixed and the maintenance of the index is centralized. These solutions, as they are, were developed for relatively static situations, and they are not suitable for the dynamic situation addressed by the proposed method, where in order to guarantee peer autonomy, peers cannot be controlled to host a certain portion of the index or to be always connected to the network; and Peers are unstable, so the structure must be capable of promptly reacting to peer disconnections, preventing hanging references in the index **[5]**.

1.3 Replication

A replication scheme capable of maintaining appropriate levels of coverage with respective to the development of user interests and network surroundings must be designed to ensure accessibility and robustness.

The main contributions of this work and its organization may be summed up as follows **[6]**.

- a compression technique for building an indexed aggregate structure over a multidimensional data population, level to be distributed, and accessed across a P2P network;
- a storage model which employs additional data structures to support proficient and robust query answering over compressed data in an unstructured P2P network ; and
- a dynamic replication idea capable of maintaining appropriate levels of coverage with respective to the evolution of the query workload and the network conditions with proposed work.

2. COMPRESSION AND INDEXING DATA

This section consists of three subsections that are partitioning, compression, and indexing.

2.1 Partitioning

The plan of the partitioning step is to divide the data area into non overlapping blocks. These blocks will be compressed separately, yielding distinct sub synopsis. For each of them, a portion of the amount of storage space B chosen to represent the whole synopsis will be invested. The distribution of B among blocks will take into account the following requirements. B must be fairly distributed among blocks and each block must be assigned a "small" portion of B. The task of different amounts of storage space to the blocks for representing their sub synopsis should depend on the differences in homogeneity among the blocks.

Intuitively enough, the more homogeneous the data inside a block, the smaller the amount of information needed to effectively achieve its summarization. The sub synopsis over the blocks is the data that will be hosted by peers and exchanged across the P2P network as shown in the Figure 1. The building of sub Synopsis with "large" size would enforce a significant constraint on the amount of storage space which should be made available by each peer **[7]**. On the converse, defining small-size sub synopsis results in limiting the storage and computational resources required at each peer for storing and querying data, as well as reducing both the download and upload traffic needed for supporting data exchange.

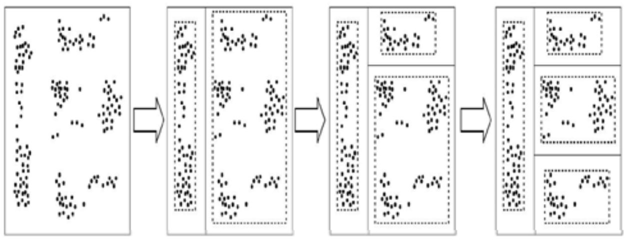


Figure 1: Partitioning a 2D data population

2.2 Compression

Clustering-based Histogram develops a density-based clustering algorithm to construct a set of blocks covering the nonempty portions of the data field **[8]**. For each block its boundaries as well as some aggregate value summarizing its data are stored. In the current implementation, each bucket is connected with the result of evaluating the sum aggregate operator. This way, the summary data suffice to estimate range sum queries.

2.3 Indexing

At this point, an index is built on top of the sub synopsis resulting from the compression step. This index will be developed for locating the data involved in the queries across the network **[9]**. The aggregate R-tree indexing the sub synopsis will be denoted as I as shown in the Figure 2.

.

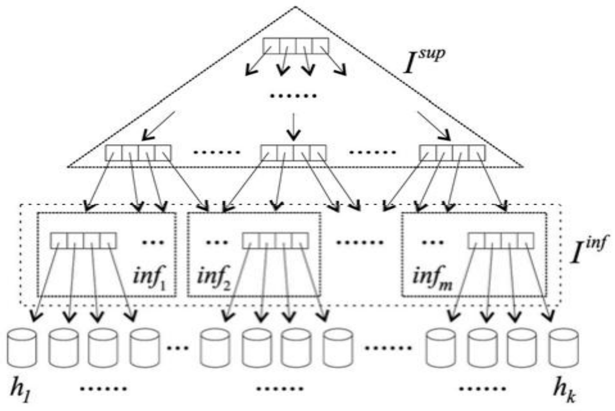


Figure 2: Partitioning the R-Tree

3. PROPOSED METHOD

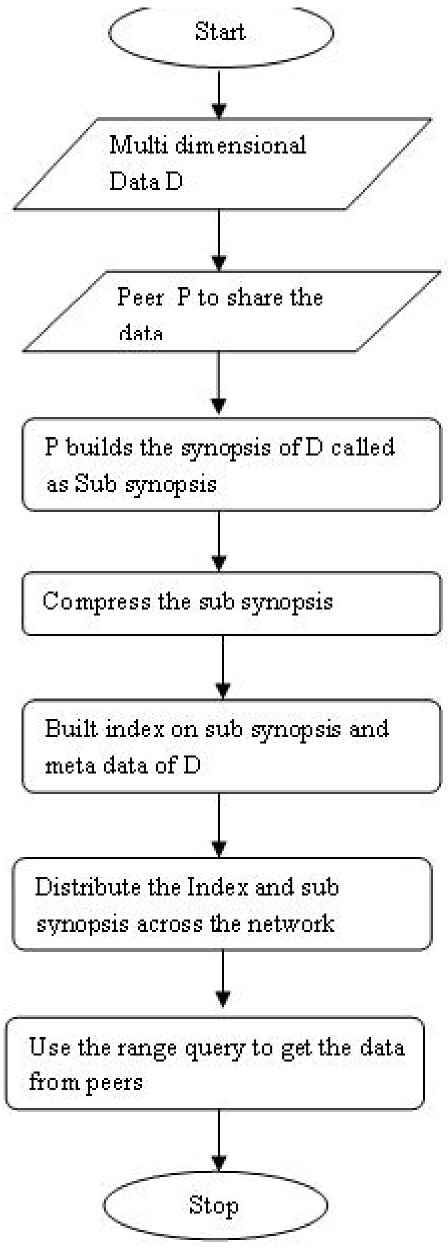

Start
Multi dimensional Data D
Peer P to share the data
P builds the synopsis of D called as Sub synopsis
Compress the sub synopsis
Built index on sub synopsis and meta data of D
Distribute the Index and sub synopsis across the network
Use the range query to get the data from peers
Stop

Figure 3: Proposed Method Flow Chart

The aim is devising a P2P-based structure supporting the analysis of multidimensional historical data. Specifically, our efforts will be devoted to combining the facilities of P2P networks and data compression to provide a support for the assessment of range queries, perhaps trading off efficiency with accuracy of answers. The structure should enable members of an society to cooperate by sharing their resources to host data and perform aggregate queries on them, while preserving their independence.

A framework with this uniqueness can be useful in different application environments. For example, consider the case of a worldwide virtual organization with users interested in geographical data, as well as the case of a real institution on an enterprise network. In both cases, even users who are not continuously interested in performing data analysis can make a part of their resources available for supporting analysis tasks needed by others, if their own capability of performing local tasks is preserved **[10]**. This is analogous to the idea on which several popular applications for public resource computing are based in order to make participants really independent, they should be imposed no constraint on storage and computational resources to be shared, as well as on the reliability of their network connection. These requirements make conventional distributed structures unsuitable and suggest the implementation of a solution based on an unstructured P2P network, where peers are neither responsible of coordination tasks, nor imposed to host specific pieces of data.

Nowadays, they are the most widespread approach for exchanging data among large communities of users in the file sharing context specifically, no P2P-based solution has imposed itself as an effective evolution of traditional distributed databases **[11]**. This is quite surprising, as the huge amount of resources provided by P2P networks could efficiently support data management. Our aim is developing a P2P-based structure supporting the analysis of multidimensional historical data. The multidimensional data is stored in peer so that it can be shared in the network, for that built the synopsis. The synopsis is built in three steps

1. Partition 2. Compressing and 3.Indexing.

The aim of the partitioning step is to divide the data domain into non overlapping blocks **[12]**. These blocks will be compressed separately, yielding distinct sub synopsis.

An index is built on top of the sub synopsis resulting from the compression step Index and these sub synopsis are distributed across the network .Queries can be created against the data. The queries can be any explorative queries. One of the first works dealing with the difficulty of supporting range queries in a peer-to-peer network is where data are ordered according to Hilbert curves, and then, distributed among the peers. The compression and indexing processes result in a synopsis organized into sub synopsis, and a fragmented aggregate R-Tree over them. The distribution of the synopsis and the index are performed.

3.1 System Primitives and Data Structures

The existence of two system primitives named search and send. Primitive search (N)—which is used by the structure every time it is required to find sets of peers on the network—returns a set of ‘N’ IP addresses of randomly chosen peers. In order to choose a peer arbitrarily, it suffices to locate a peer by starting a random walk of length rather than $\log_f N$ (where N is the number of peers in the network and f is the average fan-out) from the peer which invoked search. In fact, a random walk of this length makes the probability of reaching any peer converge to a stationary distribution, which is uniform if the network graph is well connected. In our prototype, the length of the random walk to 1’ is set. This allows us to randomly select peers from a network of up to $4^{1'}$ peers even in the pessimistic case that the network reaches a condition with average fan-out equal to 4. Primitive send(P; o) transmits s-block o from the peer p which invoked the primitives to the peer whose IP addresses are in set P. In our prototype, this primitive properly avoids overloading p when P is large. This is achieved through decentralized dissemination. Instead of sending jPj copies of o, p sends o to a subset of the peers in P which, in turn, keep a copy of o and forward it to different subsets of the remaining peers in P, and so on. We assume that each s- block is uniquely identified throughout the system, and we denote their identifiers as id (I^{sup}), id (inf_i), and id(h_j) **[13]**. Moreover, when needed to avoid confusion, we denote the s-blocks related to a population D as D.I^{sup}, D.inf_i, and D.h_j. Finally, we assume that each s-block carries along metadata about the population it belongs to. These metadata are denoted as Dm and comprise the name of the population, the schema of the data (dimensionality, names, and ranges of dimensions), as well as some keywords which will be exploited to support search operations across the network. The proposed distribution scheme makes use of a set of data structures named as location tables. Each location table will be associated with a copy of an index portion and maintain correspondences between s-blocks and sets of peers. Specifically, the location table associated with I^{sup} will consist of a row for each leaf portion, plus a row for I^{sup} itself. Each row, in turn, will contain addresses of peers where copies of these index portions are hosted. This way, a peer hosting I^{sup} will be able to contact the peers hosting copies of the leaf portions by simply accessing its associated location table. The row for I^{sup} is employed to connect the set of peers that initially host copies of I^{sup} in a clique, i.e., each peer hosting a copy of I^{sup} knows the other peers which are assigned I^{sup} as well. This way, the survivability of populations can be tightly controlled through a mechanism that replaces a peer of the clique as soon as it exits the system. Further details will be provided in the following.

In a location table associated with a copy of a leaf portion inf_i, each row will contain the addresses of the peers hosting copies of a sub synopsis pointed by inf_i . The location tables associated with index portions as table (I^{sup}) and table (inf_i) are denoted. At runtime, the local copies of these tables can be modified by the peers that host them; hence, when needed to avoid confusion, we will denote the tables at a peer p as p.table(I^{sup}) and p.table(inf_i).In addition, along with each sub synopsis and leaf portion, the address of one of the peers that point to it is stored. These reverse pointers allow for more efficient location of the peers involved in the query evaluation Process.

3.2 Disseminating Data and Index

The distribution process is started by a peer p that is willing to publish a data population, and works as follows. First, for each sub synopsis hj (respectively, leaf portion inf_i), p invokes search (C_{min}) to find C_{min} peers which can host a copy of h_j (respectively, inf_i along with table (inf_i)). Then, for each inf_i and sub synopsis h_j referenced by inf_i, location table table (inf_i) is filled with the IP addresses of the peers which will host h_j. Likewise, each h_j is augmented with a reverse pointer to one of the peers which will host inf_i. A similar process is performed to find C_{min} peers which will host I_{sup} along with a location table, and to fill the table as well as the reverse pointers of leaf portions. In meticulous, as explained before, the location table of each peer that will host a copy of I_{sup} is filled with the addresses of the other peers which will host copies of I_{sup}. After all of the location tables have been filled, the copies of s-blocks along with their associated location tables are sent to the appropriate peers. It is worth noting that distributing the copies of the s-blocks randomly across the network well suits the search of data in our unstructured scenario, where search will be performed by randomly navigating across the network. At the same time, the information provided by the location tables allows, once an s-blocks related to a data population D is located, to quickly locate all the other s-blocks that are needed to answer queries over D.

4. RESULTS AND DISCUSSION

Several experiments are performed to review the effectiveness of current approach. Specifically, the accuracy of query estimates and the performance of our copy management strategies in terms of generated network traffic; data reach ability, and query performances are studied.

4.1 Dynamic Replication

The dynamic replication idea aims at both providing the appropriate coverage of s-blocks and balancing the load at the peers. To this intend, besides guaranteeing a minimum coverage for each s-block, replication scheme provides adaptively to the dynamic query workload by creating new replicas of an s-block each time it is queried and by removing less queried data through suitable aging policies. In our structure, location tables encode links among s-blocks spread over the network. Thus, they are kept updated with respective to events causing data unavailability by deleting the addresses of the peers that no longer host these data. The present approach is independent of the way the unavailability of data is identified; in practice, this can be done through periodic pinging. After the deletion of some entries in a location table, the system detects whether the minimum coverage is maintained.

4.2 Query-Based Replication

The two replication strategies, called path based (PBS) and reactive (RS) that aim at increasing the availability of most queried data, also pursuing load balancing when facing large and dynamic query workloads are described.

4.3 Range Queries

The answer of a range query is computed at the requesting peer after receiving the answers of all the (sub) queries submitted to peers hosting data blocks overlapping the query range. The cost of a sub query can be measured from two standpoints, which take into account network-and computation-related costs.

a. *Number of hops:*

This is at least 1 for a query on I^{sup}, 2 for a sub query on a leaf portion, and 3 for a subquery on a subsynopsis. These values are lower bounds, due to peer volatility, data replacement—which yield dangling references—and overloading— which triggers the unloading mechanism.

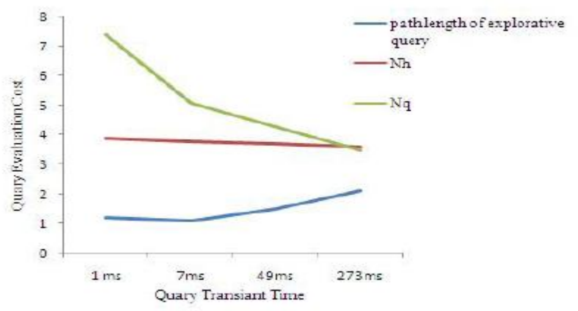
b. Overall wait in queue:

As every (sub) query SQ is enqueued at the peer p' where it will be evaluated, it has to wait for the requests preceding it. The overall wait in queue of SQ is the sum of the enqueuing position of SQ at p^1 and the overall wait in queue of the (sub)query which generated SQ (if any). For instance, if SQ is a sub query on a sub synopsis, its overall wait in queue is the sum of: 1) its enqueuing position at p';
2) the enqueuing position of the sub query SQ' which generated SQ; and 3) the enqueuing position of the query Q which generated sq'.

Thus, an upper bound on the overall time needed to complete the evaluation of a range query Q can be obtained by considering the following quantities:

N_h: the maximum number of hops performed to get the answer of a sub query of Q; and N_q: the maximum overall wait in queue for a subquery of Q.

The diagrams in Figure 4 depict N_h and N_q versus query frequency for different values of Mt(p). Fig. 4'a shows that as query frequency increases, N_h slightly increases. This can be explained as follows: in the case of PBS, a more intensive query workload yields a more frequent data replacements, which increases the likelihood of finding dangling references, and thus, of performing more hops to reach the needed data. In the case of RS, increasing query frequency causes a larger number of peers to be overloaded when they are called to evaluate queries. Thus, the unloading mechanism is triggered, and requests are forwarded to further peers, thus increasing N_h. The increase in query frequency also negatively impacts on N_q (Fig. 4'b). This effect is less evident with PBS, as compared to RS, the higher coverage allows requests to be distributed among a larger number of peers. As expected, for both N_h and N_q, the behavior of RS depends on M_t (p), as RS saturates queues before making replications: thus, waits in queue get longer as $M_t(p)$ increases, whereas the maximum number of hops for answering a sub query decreases since the unloading mechanism, yielding the forwarding of query requests to further peers, becomes less frequent as the capacity of queues increases. The results mentioned above are summarized in the Figure 4. (in the case $M_t(p) = 4$), where the cost of explorative queries (in terms of path length per query) is taken into account as well, thus providing an insight on the overall performance of the query answering process in our framework. To summarize, on the one hand, with PBS, sub queries are more likely to be served first, and the number of hops for getting the "slowest" answer of a sub query is slightly lower. On the other hand, with PBS, explorative queries require longer walks over the network to find the needed data, and the network traffic due to the replications needed to support these performances is much larger than that required by RS, thus making RS a much preferable choice.



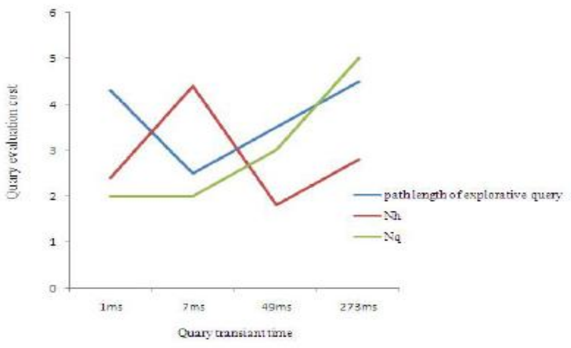


Figure 4: N_h and N_q versus query frequency for different values of Mt (p)

5. CONCLUSIONS & FUTURE WORK

As the importance of peer to peer network is increasing, the data shared in network to be stored and retrieved very efficiently. The proposed framework is to manage the multidimensional data. The data is shared and retrieval in unstructured p2p network. The people, who are interested in sharing their data, make their resources available for all peers in network. So that they can access data by posing range queries .We adopt mechanism for data summarization, data indexing and data distribution and replication by preserving autonomy of peers. The present experiment proves fast and accurate query answers and ensuring the robustness. Future work: adopting these methods to other aggregate operators rather sum. And need to devise suitable compression, indexing techniques and data distributing techniques for better robustness guarantee in the network.

6. ACKNOWLEDGMENTS

The author would like to express their sincere gratitude to the Management, Principal and Vice-Principal of LIET, Hyderabad for their constant encouragement and co-operation.

7. REFERENCES

[1] Giuseppe Massimiliano Mazzeo and Andrea Pugliese Filippo Furfaro, "Managing Multidimensional Historical Aggregate Data in Unstructured P2P Networks," *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, vol. 22, no. 9, pp. 1313-1330, September 2010.

[2] V. Poosala, and S. Ramaswamy S. Acharya, "Selectivity Estimation in Spatial Databases," in *ACM SIGMOD*, 1999.

[3] A. Andrzejak and Z. Xu, "Scalable, Efficient Range Queries for Grid Information Services," in *Proc. Second Int'l Conf. Peer-to-Peer*, 2002.

[4] M. Mihail, and A. Saberi C. Gkantsidis, "Random Walks in Peerto- Peer Networks," in *Proc. 23rd IEEE INFOCOM*, 2004.

[5] S. Chaudhuri and U. Dayal, "An Overview of Data Warehousing and OLAP Technology," *Sigmod Record*, vol. 26, no. 1, pp. 65-74, mar1997.

[6] M. Demirbas and H. Ferhatosmanoglu, "Peer-to-Peer Spatial Queries in Sensor Networks," in *Third Int'l Conf. Peer-to-Peer Computing*, 2003.

[7] http://www.gnutella.com, 2008.

[8] G. Das, D. Gunopulos, and V. Kalogeraki B. Arai, "Approximating Aggregation Queries in Peer-to-Peer Networks," in *Proc. 22nd Int'l Conf. Data Eng*, 2006.

[9] A. Gupta, D. Agrawal, and A. El Abbadi O.D. Sahin, "A Peer-to-Peer Framework for Caching Range Queries," in *Proc. 20th Int'l Conf Data Eng*, 2004.

[10] A. Krishnamurthy, and R.Y. Wang C. Zhang, "Skipindex: Towards a Scalable Peer-to-Peer Index Service for High Dimensional Data," Princeton Univ, Technical Report TR-703-04, 2004.

[11] UCI KDD Archive. http://kdd.ics.uci.edu, 2010.

[12] (2008) http://setiathome.ssl.berkeley.edu.

[13] A. Gupta, D. Agrawal, and A. El Abbadi O.D. Sahin, "A Peer-to-Peer Framework for Caching Range Queries," in *Proc. 20th Int'lConf.*, 2004.