EVALUATE DATABASE COMPRESSION PERFORMANCE AND PARALLEL BACKUP

Muthukumar Murugesan[1], T. Ravichandran[2]

[1]Research Scholar, Department of Computer Science, Karpagam University,
Coimbatore, Tamilnadu-641021, India,
amgmuthu@yahoo.com
[2]Principal, Hindusthan Institute of Technology,
Coimbatore, Tamilnadu-641032, India
dr.t.ravichandran@gmail.com

ABSTRACT

Globally accessed databases are having massive number of transactions and database size also been increased as MBs and GBs on daily basis. Handling huge volume of data's in real time environment are big challenges in the world. Global domain providers like banking, insurance, finance, railway, and many other government sectors are having massive number of transactions. Generally global providers are used to take the backup of their sensitive data's multiple times in a day, also many of the providers used to take the data backup once in few hours as well. Implementing backup often for huge size of databases without losing single bit of data is a susceptible problem and challengeable task.

This research addresses the difficulties of the above problems and provides the solution to how to optimize and enhance the process for compress the real time database and implement backup of the database in multiple devices using parallel way. Our proposed efficient algorithms provides the solution to compress the real time databases more effectively and improve the speed of backup and restore operations.

KEYWORDS

database compression, database decompression, backup optimization, real-time database

1. INTRODUCTION

The term compression refers to the process of reducing the amount of data from the original information. Database compression is extensively used in data management for performance improvement and to save storage space. Data available in a real time database are highly valuable. Data compression is the art of finding short descriptions for long strings. Every compression algorithm can be decomposed into zero or more transforms, a model, and a coder. Data compression, a fundamental aspect of today's world, is often cited as a triumph of basic research. The research behind database compression is difficult and abstract. This new research will introduce the new approach to the world of data compression. The fine contributions will help you begin to understand the wide variety of approaches to compressing textual data.

Parallel backup is process of periodically taking a copy of the database and log file to offline storage media for the future reference. This is very important for the critical database environment. This kind of daily activities will consume considerable time, volume of resources and highly expensive. Parallel backup reduces storage costs and requirements and minimizes downtime, which saves both time and money.

DOI : 10.5121/ijdms.2013.5402

Backup compression provides combination to reduce database and backup storage costs. Over the years hard disks have become very reliable but, like any other device, they are not perfect and can fail for a variety of reasons. When they do fail, it is important that the data they contain is not lost. To protect data against loss, users must store a copy of the data in some secondary location. The database files used to store the database are sorted by a disk device, and a reader thread is assigned to each device. The reader thread reads the data from the database files. A writer thread is assigned to each backup device. The writer thread writes data to the backup device. Parallel read operations can be increased by spreading the database files among more logical drives. Similarly, parallel write operations can be increased by using more backup devices. Backup compression offers the following benefits.

- Reducing Storage Requirement
- Data Transfer Rate
- Enhancing Data Security
- Backup and Recovery
- Performance Enhancement

When introducing data compression technology into real-time database, two requests must be satisfied. First, proposed algorithm should be efficient to compress the real-time database. Second, the compression algorithm must fast enough to satisfy the function of real-time record and query in real-time database.

The homogeneity of data can affect the compression radio of most compression algorithms, but it doesn't have an effect on compressed speed. In order to achieve better compression performance, the compression algorithms are specially designed for every portion of the data.

Proposed algorithms provide the solutions for the above issues and repeatedly increase the compression ratio at each scan of the database systems. These algorithms have been specifically designed to accomplish these requirements, resulting in increased application and business availability for our critical database environment.

2. BACKGROUND

During the database compression there are many chances of hurting the data in the database systems. We have to apply the new mechanism and techniques to extract the original data without even losing single bit. Database compression and decompression is a susceptible problem now a days and database compression is widely used in data management to improve the performance and save storage space. The basic idea behind multi storage parallel backup is to store the backup data using data repository model ranging from single device to multiple devices. The Parallel backup uses a full incremental repository data model which provide with higher level of feasibility for storing the backups in several devices by increasing the speed of backup and restore operations.

The compression process consists of two separate activities, modeling and coding. Modeling defines how each of the distinct symbols in the input stream will be represented. A model stores information on how often the symbol occurred in the data, that is, symbol probabilities. Coding, the second part of the compression process, results in a compressed version of the data by constructing a set of codes for the distinct symbols based on the probabilities provided by the model.

Real-time database is the combination of real-time system technology and database technology. Real-time database has the characteristic of high speed, high data throughput and so on.

Compression schemes can be classified according to the way the model alters during both compression and decompression. Importantly, new approaches permit random-access to the compressed data, allowing atomic decompression of records stored in databases. This proposed approach require two separate passes of the data: the first gathers statistics on the symbols necessary to build the model; and the second encodes the symbols according to the model.

3. PERFORMANCE EVALUATION

Over the last decades, the amount of data stored in real time database systems has grown at a very high speed. The main purpose of storage of data lies in the periodic compression of data both for authentic purpose and recovery of data in certain cases where failure of the system occurs. For many organizations, the data loss results in disastrous way. Hence, backup compression of data in database has to be performed on a regular interval of time. The structure of the proposed approaches for Database Compression Performance and Parallel Backup are shown in fig 1.

3.1 Evaluate Database Compression Performance

The compression of database systems for real time environment is developed with our proposed **Iterative Length Compression (ILC)** algorithm. This technique has enabled to improve performance by trading reduced storage space and I/O against additional CPU overhead for backup data.

ILC algorithm approach provides much lower time complexity contrast to an existing technique. It provides good compression, while allowing access even at attribute level. This algorithm repeatedly increases the compression ratio at each scan of the database systems. Compression can significantly reduce disk space storage, increase memory utilization, and better I/O performance; there is a risk of CPU degradation. Database analysis is apprehensive with the environment and use of data. It engages the classification of the data elements which are desired to sustain the data dealing out system of the organization, the introduction of these elements into rational groups and the description of the relations among the resulting groups.

Evaluate Database Compression Performance and Parallel Backup is analyzed based on the environment created. Initially, the attributes present in the database are analyzed and maintains the set of attributes and tables. Then, the analyzed databases are efficiently compressed by using Iterative Length Compression (ILC) algorithm. The ILC algorithm is used to provide a good compression technique by allowing access to the database level and enhances the compression ratio for an ease backup of database systems. The main job of the real-time database systems is to process the larger quantities of data for different purposes. The rate of increase in the processing of data has resulted in the requirements for multiple data storage systems that provide capability to massive storage of data with the inclusion of faster access for the different nodes. The other significant point possessed by many organizations is the availability of data in a continuous manner. Many organizations work on round-the-clock basis and access the database and store them in multiple locations which are also referred to as multi storage systems. The data stored in the multi storage systems are increasing at a significant rate.

3.2 Enhanced Database Parallel Backup

In real-time environment, essential to take back up of the compressed databases sporadically. This is extremely important for the critical database environment. Our proposed efficient algorithm provides the solution to improve the speed of backup and restore operations. Optimal multi storage parallel backup approach for real time database system by using Parallel Multithreaded Pipeline (PMP) algorithm to store compressed database at multiple devices in parallel. Using

multiple devices can increase throughput in proportion to the number of devices used and also parallel backup reduces storage costs, requirements and minimizes downtime.
When you use multiple backup devices is the same as creating and restoring backups when you use a single device. The only difference is that you must specify all backup devices involved in the operation, not just one. If a database backup is to be created that uses three tape backup devices such as DEVICE0, DEVICE1 and DEVICE2, each of the tape/storage devices must be specified as part of the backup operation, although fewer tape backup devices can be used when you restore the backup later.

Storage space ability and computational authority increases, processing and analyzing large volumes of database systems acts as a significant role in several areas of scientific research. Using multiple devices can increase throughput in proportion to the number of devices used. When you create a backup on multiple backup devices by using removable media, the devices can operate at different speeds and the media volumes can have different amounts of space available. Internal synchronization point occurs while any device is blocked, the backup operation pauses entirely until that device is made the available again. Multi Storage Parallel Backup is a promising technology that enables a real-time information infrastructure to provide with high-value, timely content in order to monitor and tracking applications. Multi Storage Parallel Backup infrastructure is likely to revolutionize areas such as pharmaceuticals, management, supply chain management, in the coming decade.

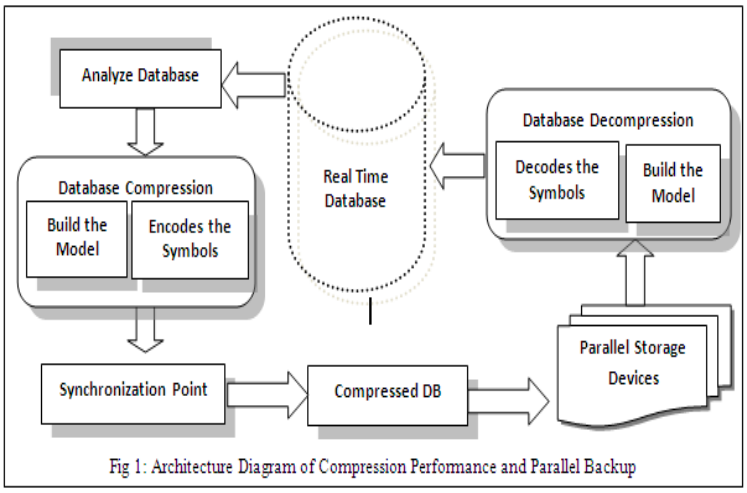

Fig 1: Architecture Diagram of Compression Performance and Parallel Backup

4. EXPERIMENT CONFIGURATION

Goal of our experiments is to measure the performance of the proposed algorithms based on real time databases. We had compared the performance of our implementation with widely available data backup algorithms running on the same hardware. We performed experimental tests on systems that correspond to the ends of the commodity hardware spectrum. The proposed approach consumes less time to backup and restore the data since it preceded the restore operation in an optimal manner. Proposed algorithm has been specifically designed to meet these requirements, resulting in increased application and business availability for our critical database environment. The experiments were tested using the below configuration.

System Component	Description
Processors	Intel® Core 2 Duo CPU P-IV 2 GHz Processer
Memory	3 GB memory
Operating System	Microsoft Windows XP Professional Version 2002 Service Pack 3
Database Manager	Microsoft SQL Server 2008 Enterprise Edition (64 bit)
Database Size	1 GB

Table 1: System Components

Data volume is not only the most important portion in the data structure, but also the least regular portion. But we can still find some rule in its data curve. The correlativity of data value is weak and there is usually small wave between two neighborhood data points. Any compression algorithm can't achieve high compression radio towards this kind of data. Considering both compression radio and speed, it is suitable to use ILC algorithm or its variations to compress historical data.

The rule of time stamp is obvious. Normally, the value of time stamp is arranged in arithmetical series. So the increment of time stamp is invariable, which is very suitable to be compressed by ILC compression algorithm. Quality code has the highest redundancy in three kinds of data. It seldom jumps and always keeps the same value, which is suitable to be compressed by ILC compression algorithm too. The test for simulation data indicates that the compression ratio of ILC algorithm for quality code can achieve 85% and the time for compression and decompression can be considered as very less.

Real-time database applied in global industry requests the performance of mass data and high speed. So the process data in database must be compressed effectively and reliably. According to process data characteristic, a lossless compression algorithm was designed based on RLE algorithm and Dictionary algorithms.

5. RESULTS COMPARISONS

In this work, we efficiently handled the database compression and stored the compressed database in multiple devices simultaneously. Using an optimal approach, real time databases have been compressed effectively and the compressed databases been stored under different devices simultaneously. Parallel operations can be increased by spreading the database files among more logical drives.

5.1 Compression Ratio and Space Savings

The below table and diagram described the compression ratios and storage space savings of the proposed backup compression process for real-time database systems using ILC algorithm.

Compression Types	Compressed File Size (MB)	Compression Ratio	Space Savings
Uncompressed	1024	1.00	0%
RLE Compression	580	1.77	43%
Dictionary Compression	525	1.95	49%
ILC Compression	370	2.76	64%

Table 2: Compression Ratio vs Space Savings

The above table 2 described the compression ratio and space savings based on size of data present in the database. The efficiency of compression using the proposed backup compression process for real-time database systems using ILC algorithm is compared with an existing compression algorithms. The compression performance has been tested here is compared with the general compression module. The tested data is from simulation data is based on historical data characteristics. Below graph shows proposed ILC algorithm achieves less storage space and the variance would be about 30% to 60% better.

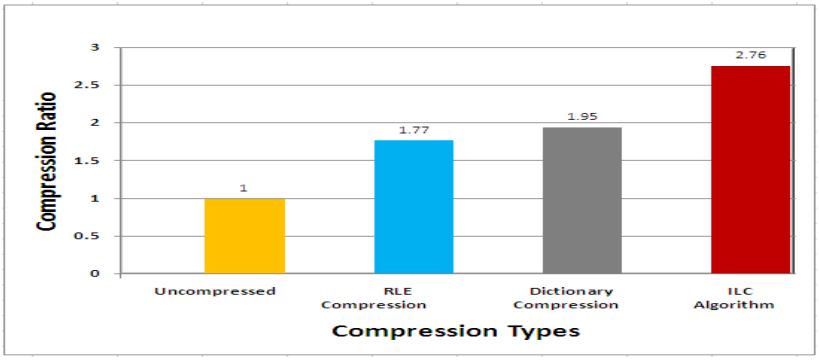

Fig 2: Compression Types vs Compression Ratio

Fig 2 describes the process of compression ratio based on different types of existing compression algorithms. The compression ratio becomes less in the proposed backup compression process for real-time database systems with parallel multi-storage process. The compression ratio is measured in terms of megabyte (mb).

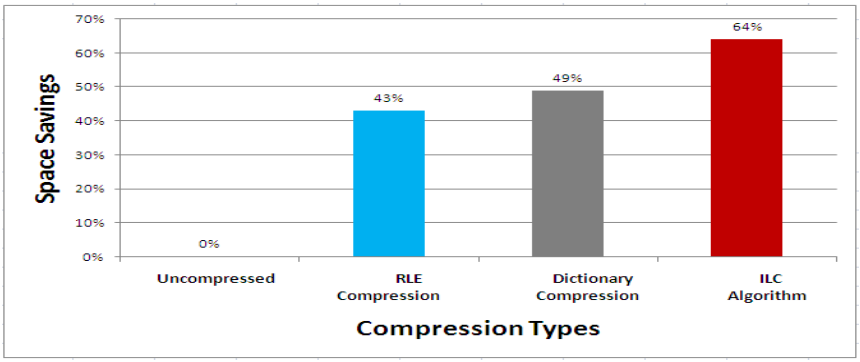

Fig 3: Compression Types vs Space Savings

Fig 3 describes the process of space savings based on different types of existing compression algorithms. The space savings is measured in terms of MBs. Compared to an existing compression algorithm, the proposed ILC algorithm achieves less storage space and the variance would be about 60% better.

Proposed compression significantly decreases disk storage and backup/restore times also been greatly reduced. We experiment the data base size ranging from relatively simple to fairly complex. The most obvious advantage of database compression is that of reducing the storage requirement of information. Reducing the storage requirement of databases is equivalent to increasing the capacity of the storage medium.

5.2 Analysis of Memory Gain

Compressing large database consume significant I/O volume, improve their memory caching and reduce the I/O volume enough to compensate for the compression/decompression overhead, thus reducing storage costs without undue change in performance. In certain I/O-bound situations, data compression can even improve overall performance. For transaction processing, there will probably be two main effects of compression. First, the buffer hit rate should increase since more records fit into the buffer space. Second, system memory should effectively utilize to gain the performance. The results below show performance of gain in memory and storage utilization.

Compression Types	Uncompressed Database	Existing attribute level compression	ILC Compression
Memory Utilization	10381 K	8480 K	7286 K
Memory Gain	0 %	18 %	30 %

Table 3: Compression Types and Memory Utilization

Below Figure4 shows the memory usage for large database. As in the above figures, even a small amount of compression improves the performance significantly. Generally analyzing the database performance can be divided into three sections on I/O performance and buffering, transaction processing, and query processing. Below figure shows the memory usage for large database. As in the above figures, even a small amount of compression improves the performance significantly.

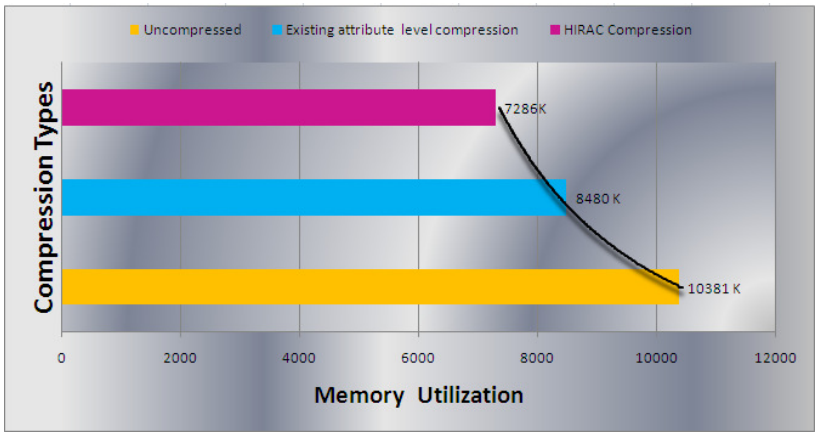

Fig 4: Compression Types and Memory Utilization

5.3 Analysis of Backup Time

The CPU overhead in the proposed Multi Storage Parallel Backup is low. The below table described the time taken to backup the data in multiple storage devices simultaneously. The efficiency of multi storage in parallel using the optimal multi storage parallel backup data compression for real time database systems of size 1 GB is compared with the existing simple model for storing database at multiple devices in parallel [SMSD].

No. of Data (MBs)	Time to Backup (sec)	
	Proposed MSPB	Existing Method
25	5	15
50	13	20
75	9	29
100	24	25
125	20	33

Table 4: Database Size and Backup Time

Below figure described the processing time to take backup of compressed data into multiple storage devices in parallel. The MSPB using Parallel Multithreaded Pipeline algorithm consumes less time to backup and restore the compressed data. In the proposed MSPB, the restoring of the compressed data with the devices is performed with optimizing the parallel multi storage devices. If the number of data in the database increases, the time to restore all the compressed in the storage devices simultaneously is less.

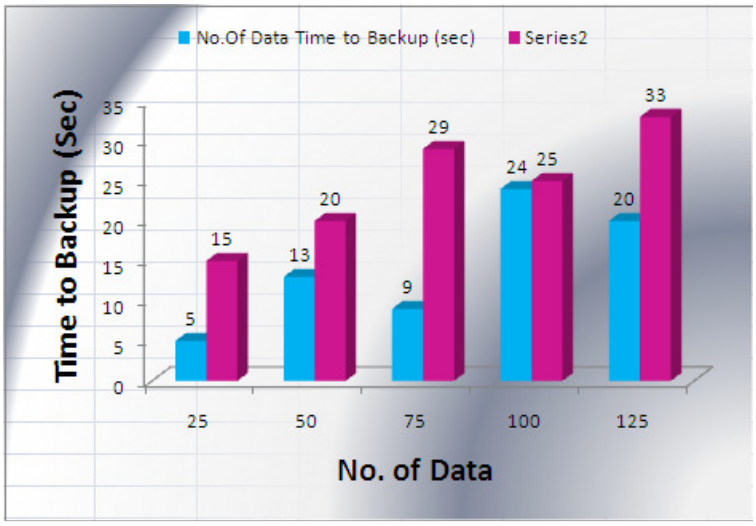

Fig 5: Database Size and Backup Time

Since the optimization technique is used with the parallel multi storage devices, the MSPB consumes less time to backup the data. The backup time is measured in terms of seconds (sec). Compared to the existing simple model for storing database at multiple devices in parallel, the optimal multi storage parallel backup data compression for real time database systems consumes less restoring time and the variance is more than 35%.

6. CONCLUSIONS

Proposed compression approach will allow efficient retrieval of the compressed data, as well as produce a saving in storage costs. Additionally, the compression scheme must be able to independently decompress a random selection of records. The work we have reported here shows that this is substantially possible, by appropriate trade-offs of performance versus degree of dynamic reorganization.

This paper introduces a compression algorithm used in real-time database, which designs suitable compress method for every kind of historical data after synthetically considering compression radio, space and speed. Compared with other compression algorithms that have been used in real-time database, its advantage is that it realizes lossless compression. We can see from the result of performance test that the compression algorithm has high compression performance and possesses great application value. The main aim of this research is to increase total throughput of the database compression. This research improves the performance of conventional database compression and parallel data storage systems by using an innovative data transfer process.

Proposed compression also improves CPU performance by allowing database operators to operate directly on compressed data and provides good compression of large databases. This paper shows that implementing lightweight compression schemes and operators that work directly on compressed data can have significant database performance gains. In overall besides the contributions of the introduction of new algorithms improved the current state of the art and most suitable for real-time environment.

REFERENCES

[1] Gupta, A., "Nonlinear Sparse-Graph Codes for Lossy Compression", Information Theory, IEEE Transactions on, Volume: 55, Issue: 5. 2009.

[2] Adam Cannane , Hugh E. Williams "A Compression Scheme for Large Databases" proceedings in 11th international conference on ADC, 2010.

[3] D. J. Abadi. Query execution in column-oriented database systems.MIT PhD Dissertation, 2008. PhD Thesis.

[4] Minos N. Garofalakis., Rajeev Rastogi. Spartan and Shivnath Babu., "A model-based semantic compression system for massive data tables. In SIGMOD", 2001.

[5] W. P. Cockshott, D. McGregor, N. Kotsis, J. Wilson "Data Compression in Database Systems", proceedings in 9th international workshop on Database and Expert Systems Applications, 1998.

[6] Chenggang Zhen , Baoqiang Ren, "Design and realization of data compression in Real-time database", proceedings in 10th international conference on Computational Intelligence and Software Engineering , 2009.

[7] Veluchandhar, RV.Jayakumar,M.muthuvel,K.Balasubramanian,A.Karthi,Karthikesan, G.Ramaiyan, ,A.Deepa.S.AlBert Rabara, "A Backup Mechanism with Concurrency Control for Multilevel Secure Distributed Database Systems", proceedings in 3rd international conference on Digital Information Management (ICDIM) , 2008.

[8] Hung-Yi Lin and Shih-Ying Chen, "High Indexing Compression for Spatial Databases", proceedings in IEEE 8th international conference on Computer and Information Technology Workshops, 2008.

[9] Ioannis Kontoyiannis and Christos Gioran "Efficient Random Codebooks and Databases for Lossy Compression in Near-Linear Time", proceedings in IEEE Information Theory Workshops on Networking and Information Theory, 2009.

[10] Computer Society Press, Los Alamitos, California. A. Cannane and H.Williams. General-purpose compression for efficient retrieval. Technical Report TR-99-6, Department of Computer Science, RMIT, Melbourne, Australia, 1999.