International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Management of Large Scale Data

Neha Upadhyaya

M.Tech.(C.S.E)

Abstract: ***Storage management is important for overall performance of the system. Slow access time of disk storage is crucial for performance of large scale computer systems. Two important components of storage management are buffer cache and disk layout management. The buffer cache is used to store disk pages in memory to speed up the access to them. Buffer cache management algorithms require careful hand-tuning for good performance. Aself-tuning algorithm automatically manages to clean the page activity in the buffer cache management algorithm by monitoring the I/O activities. Buffer cache management algorithm is used for global data structure and it is protected by lock. Lock can because contention is helpful in reducing the throughput of the multi-processing system. Currently used solution for eliminating lock contention will directly degrade hit ratio of the buffer cache will result in poor performance in the I/O bound. The new approach, multi-region cache eliminates the lock contention without affecting the hit ratio of buffer cache. Disk layout approach improves the disk I/O efficiency, called Overwrite, and is optimized for sequential me /so from a single file. A new disk layout approach, called HyLog. HyLog achieves performance comparable to the best of existing disk layout approaches in most cases.***

Keywords: Buffer cache management, performance of large scale computer systems, lockcontention, Different disk layout management approaches

1. Introduction

Database management system (DBMS) is the important part for the development of internet and other large scale systems. Our existence approach of DBMS i.e. rational and object oriented DMBS has a problem of flexible scaling required for modern systems. These huge databases could no longer be contained in one physical system, but should be run on a distributed system. The CAP theorem states that it is impossible for a distributed computer system to simultaneously provide all of consistency, availability, and partition tolerance. To overcome these limitations a new approach of NoSQL databases has started. Structured query language (SQL) is a programming language designed for relational DBMS. NoSQL databases tend to embrace the concepts of high availability and eventual consistency. Storage is an important part of all large computer systems.

Storage is often the performance bottleneck of the system, especially for large scale systems accessing large amounts of data, such as storage servers, fileservers, web servers, email servers, and database servers. Storage servers, such as the EMC Symmetric series [1], provide disk storage to other systems. Storage servers often provide raw storage services to file servers and database servers. FAS series of file servers of NetApp Inc. [2], provide file services to other systems. Storage management is used in these systems to speed up accesses to data on disks. Storage management employs an in-memory buffer to cache popular disk pages and also manages how data are placed on disks. The part of the system controlled by storage management is called the storage subsystem, whose performance is often crucial to the performance of the whole system. The goal of this review research is to investigate techniques to improve the performance of storage subsystems in large scale systems. Database workload is an important class of workload in the storage subsystem. Database systems can be divided into two categories: on-line transaction processing (OLTP) and decision support

.

1.1 Motive of this Paper

This thesis studies the storage subsystem on large scale systems, addressing several performance issues of the buffer cache layer and the disk layout layer. This includes analyzing the characteristics of typical workloads running on large scale systems, identifying the performance bottleneck of the storage subsystem, modeling and simulating key components of the storage subsystem and designing and evaluating new algorithms which can ease the tuning task or achieve better performance. Direct experimentation, trace-driven simulation, and mathematical modeling are used in this study. The main contributions of the thesis are:

- A new algorithm is proposed to automatically tune parameters of buffer cache management in order to achieve good performance. Its effectiveness is comparable to the best manually tuned algorithm.
- The problem of lock contention in buffer cache management is investigated. A new approach called the multi-region cache is proposed to eliminate the lock contention of the buffer cache. This approach can work together with most buffer cache replacement algorithms. It does not compromise the overall hit ratio of the buffer cache and incurs little overhead.
- Different disk layout management approaches are modeled and their performance characteristics are analyzed. A new approach called HyLog is proposed. HyLog achieves performance close to the best of existing approaches in most configurations.

1.2 Self-Tuning Of Buffer Cache Management

A buffer cache management algorithm in a real system often has many parameters that need to be tuned for the particular workload and system configuration at hand. However, such tuning is often not easy. The buffer cache management algorithm in IBM's DB2 database management system employs page cleaners to write the dirty pages in the buffer cache asynchronously so that Lock Contention of Buffer Cache Management.

Most buffer cache replacement algorithms use a global data structure to manage all pages in the buffer cache. In large scale systems with many processors and threads, different threads may access the buffer cache simultaneously. The global data structure of the replacement algorithm must be protected by a lock to avoid corruption. This lock may become the contention point (called lock contention) and limit the system throughput. Real systems solve the contention problem in different ways; depending on their particular requirements.SQL Server 7.0 uses a CLOCK-based algorithm to reduce contention [3], since this kind of algorithm does not modify the global data structure on buffer cache hits. However, a CLOCK-based algorithm typically has a lower hit ratio than other replacement algorithms and may cause poor system performance. Berkeley DB and ADABAS use different variations of LRU without global data structures but these approaches either have high overhead or cannot be applied to other replacement algorithms. These current practices in buffer cache management motivate the search for a better approach to reduce lock contention without compromising the overall hit ratio. A new buffer cache management approach, called the multi-region cache, is proposed for this purpose. The multi-region approach divides the buffer cache into many fixed-size regions, each of which is managed by an instance of a replacement algorithm. Each page is mapped into a unique region. Since lock contention happens only when pages within the same region are accessed at the same time, and there are a large number of regions, lock contention almost never happens in the multi-region cache. Both analysis and simulation results show that a large buffer cache with hundreds of thousands of regions has almost the same overall hit ratio as the traditional approach, and the overhead is negligible. Multi-region cache can be applied to most replacement algorithms.

1.3 Disk Layout Management

With a large buffer cache, most disk reads can be resolved in memory [4].As a result, in many systems, write requests make up a large portion of the total traffic to the disks. These write requests are from many users and often scatter over the disks, which results in low utilization of the disk transfer bandwidth when the disk layout is managed by the commonly used Overwrite approach.LFS was designed to provide good write performance while maintaining comparable read performance in such systems, but the high segment cleaning overhead of LFS decreases its performance dramatically.

The write performance of Overwrite and LFS is modeled and the impact of changing disk technology on their performance is investigated. Because of the much faster improvement in disk transfer bandwidth than disk positioning time, it is found that LFS significantly outperforms Overwrite under modern and future disks over a wide range of system configurations and workloads.LFS performs worse than Overwrite, however, when the disk space utilization is very high because of the high segment cleaning cost. A new approach, the Hybrid Log-structured (Hilo) disk layout, is proposed to overcome this problem. HyLog uses a log-structured approach for hot pages to achieve good write performance, and Overwrite for cold pages to reduce the segment cleaning cost. An adaptive separating algorithm is designed to separate hot pages from cold pages under various workloads and system configurations. This algorithm works in real time and incurs little overhead. Simulation results under a range of system configurations and workloads show that, in most cases, Hilo performs comparably to the best of the existing disk layout approaches.

2. Methodology

2.1 Typical Workloads

Understanding the basic workload characteristics of the storage subsystem is a prerequisite to studying its performance. The workloads presented to the storage subsystem can be classified roughly into database workloads and file server workloads. Since storage servers often sit below other applications which already employ large buffer caches, such as database servers or file servers, the requests to a storage server may already have been filtered by upper layer buffer caches, and so may exhibit different characteristics. These three categories of workloads are discussed separately in the following subsections.

2.1.1 Database Workloads

Database servers support a wide range of applications. They can be classified roughly into two classes: on-line transaction processing (OLTP) systems and decision support systems (DSS). DSSs are also called on-line analytical processing (OLAP) systems. Real database applications have the properties of both [5].As a new way of doing business through the Internee-commerce applications play an increasingly important role in database applications. The database workloads in e-commerce applications can be viewed as a mix of OLTP and decision support workloads [6].

2.1.1.1 OLTP Workloads

OLTP workload is important in large database systems. OLTP applications are used in many industries for data entry and data retrieval transactions. OLTP is the cornerstone by which a great deal of modern business is done. Most OLTP transactions are quite simple. The execution time of each transaction is short (typically within a second),and there are upper bound requirements for response time. An OLTP application has many terminals connected to one or more central database servers through a network as shown in Figure 2.1.The client/server model is typically used for OLTP applications. Different terminals initiate various transactions to the server independently. The database on the server is updated frequently by these transactions. These updates are typically small and to random places of the disks. Because of the I/O-intensive nature of OLTP applications, the storage subsystem is often the performance bottleneck. The two major design challenges for the storage subsystem to achieve good performance for OLTP applications lie in the buffer cache layer and the disk layout layer. The buffer cache of the DBMS must be managed effectively to reduce the number of disk accesses. Since most reads are absorbed by the buffer cache, random updates make up a large proportion of requests to disks. The disk layout must be organized to handle the random updates

efficiently. The TPC-C benchmark [7], which is a standard benchmark representing OLTP workloads, is used in this thesis to study the performance of storage management under OLTP workloads.

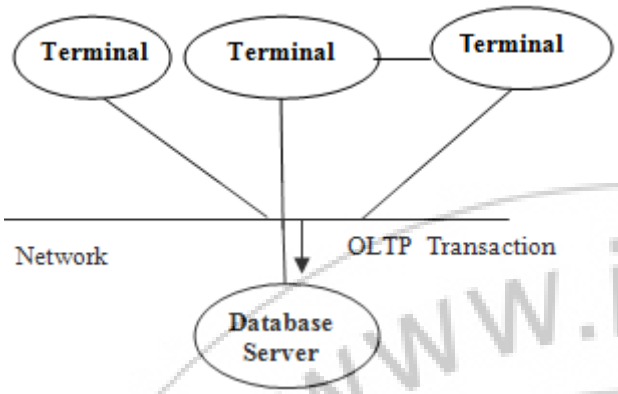

Figure 2.1: Architecture of an OLTP Application

2.1.1.2 Decision Support Workloads

In a DSS, users query business data to get answers to critical business questions. In decision support workloads, complex queries are used to search a large amount of data in the database. The execution time of each query is long (from tens of seconds to several hours), and the queries mainly read the database sequentially.

2.1.1.3 E-commerce Workloads

E-commerce is a new way of selling products or services through the Internet. Customers access an e-commerce web site from their web browsers. A typical e-commerce system contains a back-end database server and several front-end servers, including web servers, web caches, and image servers. Figure 2.2shows a typical structure of an e-commerce environment. The web servers provide web pages to browsers. The image servers provide images to browsers. The web cache servers cache the search results of client requests to reduce the load on the database server. The back-end database server stores the information of customers and products and processes user transactions.

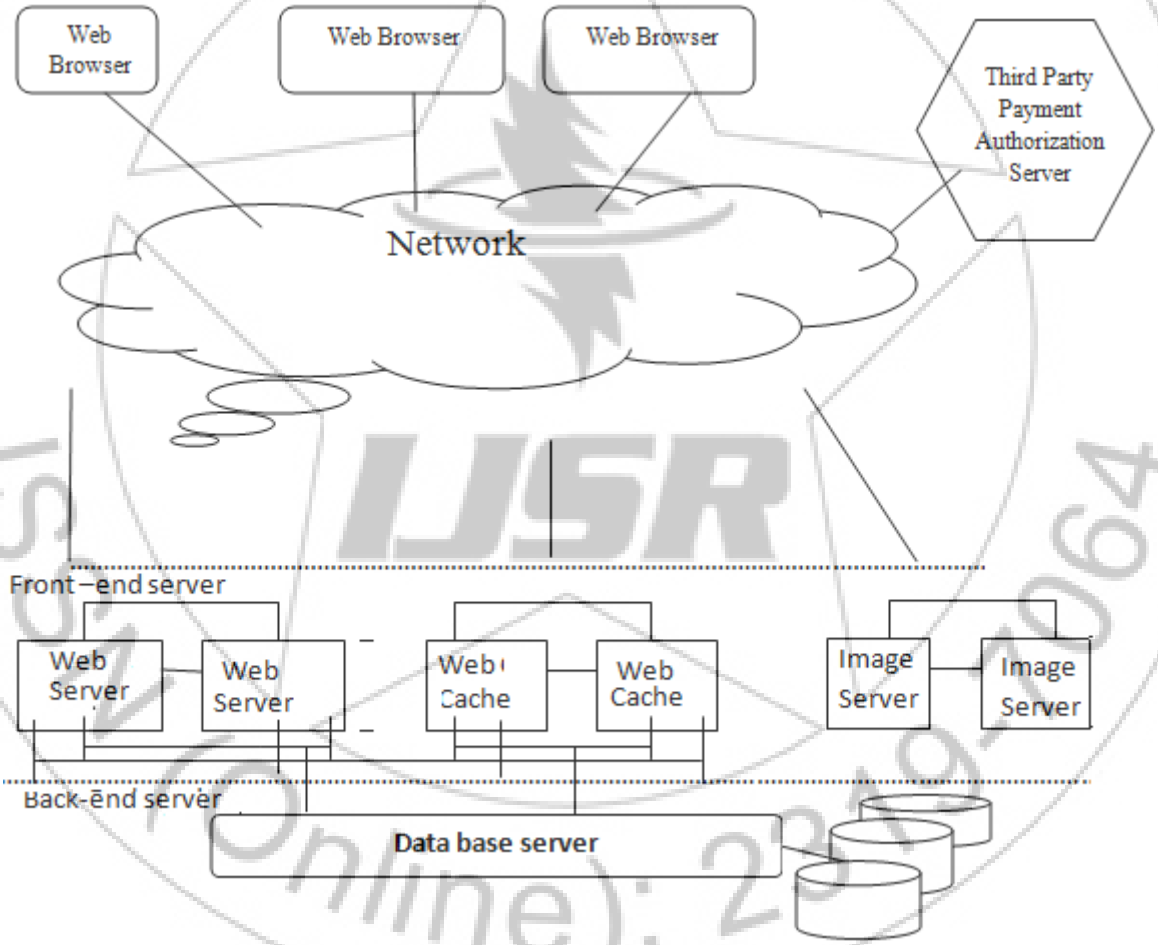

Figure 2.2: E-Commerce Environment

2.2 Tpc-C Workload Characterization

The TPC-C benchmark models the order processing operations of a wholesale supplier with some geographically distributed sales districts and associated warehouses. The business environment is illustrated in Figure2.3.In the TPC-C benchmark; the number of warehouses is a variable which determines the scale of the benchmark. Each warehouse has 10 sales districts and each district serves 3000 customers. The supplier has 100,000 items for sale. The initial size of one warehouse is about 100M bytes data.

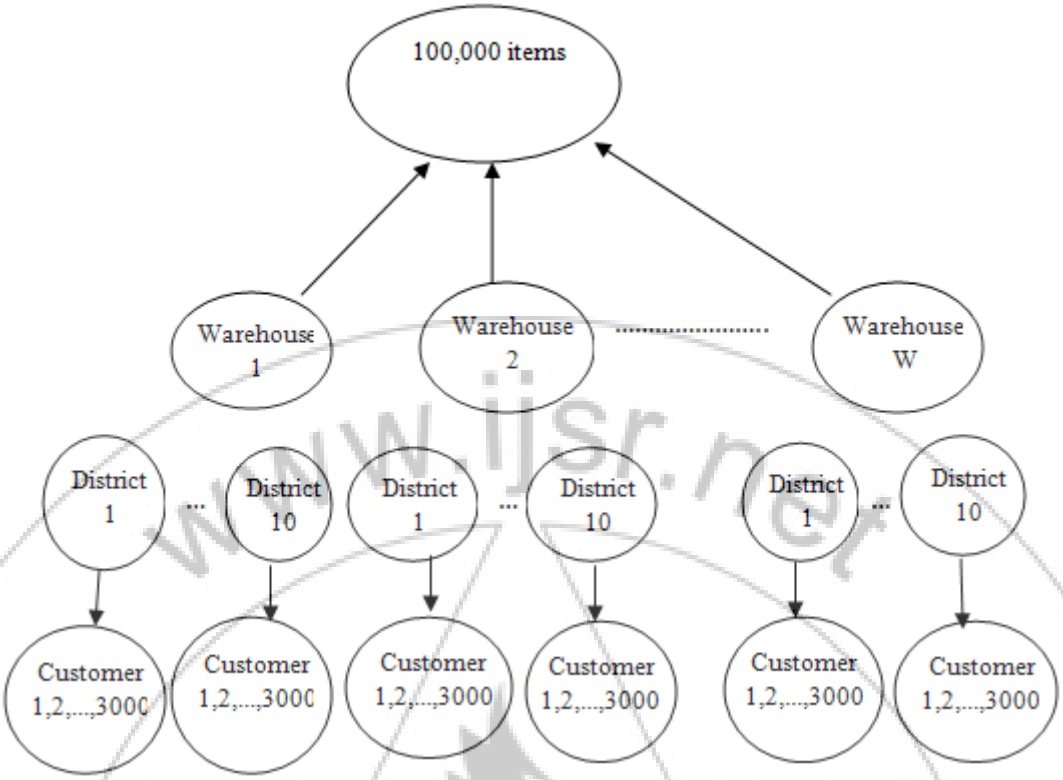

Figure 2.3: TPC- Business Environment

3. Future Work

There are many open research issues related to this thesis work. Some possible future research directions may include:

[1] Studying whether the self-tuning page cleaning algorithm can respond well to workload changes in the system.
[2] Implementing the self-tuning page cleaning algorithm in a real DBMS system and evaluating its performance under real workloads.
[3] Implementing and evaluating the multi-region cache approach in real systems.
[4] Improving the multi-region cache so that its performance is more stable when a large number of regions are used. Simulation results indicate that the hit ratio of the multi-region cache decreases when there are too many regions. This is because some regions have too low hit ratios. If these regions can be identified and their pages can be rearranged, multi-region cache could
[5] Achieve good performance even when many regions are used.
[6] Designing new cache replacement algorithms which can take advantage of full region scan at cache misses using multi-region cache. Scanning the whole region on a cache miss incurs little overhead when each region is small. This relaxes the time complexity from O (1) to O (n), and thus enables more design choices when designing replacement algorithms.

References

[1] EMC Corporation
[2] Network Appliance Inc.
[3] Microsoft Corporation. "Microsoft SQL Server 7.0 storage engine capacity planning tips". MSDN Library, March 1999.
[4] Dave Anderson, Jim Dykes, and Erik Riedel." More than an interface | SCSIvs. ATA".
[5] Windsor W. Hsu, Alan Jay Smith, and Honesty C. Young. "Characteristics of production database workloads and the TPC benchmarks."
[6] Said S. Ulnar. "A methodology for auto-recognizing DBMS workloads".
[7] Transaction processing performance council.
[8] Kevin W.Froese and Richard B. Bunt. "The effect of client caching on file server workloads".
[9] Dominic Giampaolo. "Practical File System Design. Morgan Kaufmann, 1999."
[10] David Jacobson and John Wilkes. "Disk scheduling algorithms based on Rotational position".

Author Profile

Neha Upadhyaya is pursuing Match from Department of Computer Science & Engineering World College of Technology And Management& Affiliated To Maharshi Dayanand University,Haryana, India