CHAPTER 1: INTRODUCTION

1 Introduction

Chapter Snapshot

- Description of buffer, buffer types, buffer partitions
- Description of buffer management in ODBMS
- A need of effective and efficient buffer management
- Evolution of buffer management in RDBMS
- Main goal and work organization of my research.
- Description of previous work and literature.
- Chapter wise organization of my thesis

1.1 Overview

Database management systems (DBMSs) use external magnetic devices (disks) for the storage of mass data. They offer low cost per bit and non-volatility, which makes them indispensable in today’s DBMS technology. However, under commercially available operating systems, data can only be manipulated (i.e., compared, inserted, modified, and deleted) in the main storage of the computer. Therefore, part of the database has to be loaded into a main storage area before manipulation and written back to disk after modification. A database buffer has to be maintained for purposes of interfacing main memory and disk. Although several modern operating systems provide a main storage “cache” for their file systems, most DBMSs have their own buffer pools in the user address space—they do not use the OS file cache. In order to facilitate the exchange of data between disk and main storage, the database is divided into pages of equal size (generally 512 to 4096 bytes). The buffer consists of page frames of the same
size. The number of frames in the buffer can be selected as a DBMS parameter, which remains constant during a DBMS session. Today buffer sizes vary from about 16 K to 12 M bytes. A typical buffer size may be assumed to be between 128 K and 256 K bytes.

Since a physical access to a database page on disk is much more expensive than an access to a database page in the buffer, the main goal of a database buffer manager is the minimization of physical I/O for a given buffer size. This goal has to be accomplished under certain restrictions resulting from the interface between the buffer manager and other DBMS components. Various reasons (for a Buffer allocation algorithms can be subdivided into local and global algorithms. An algorithm is local if it allocates buffer frames to a specific transaction without regarding the reference behavior of concurrent transactions. For a database buffer manager, local algorithms have to be supplemented with a mechanism for handling the allocation of buffer frames for shared pages, since concurrent access to the same database page is frequent. Local algorithms can be further subdivided into static and dynamic allocation. Under static allocation, the number of buffer frames belonging to a transaction remains constant during the lifetime of the transaction. A simple algorithm is the allocation of a fixed-size partition to each of the parallel transactions; more sophisticated algorithms could assign partitions of different sizes to different types of transactions, based on information known at the start time of a new transaction. Dynamic allocation assigns variable-size partitions to the transactions; each partition can grow and shrink according to the current reference behavior of the transaction. In contrast to local algorithms, global allocation algorithms consider not only the reference pattern of the transaction currently executing, but also the reference behavior of all other transactions. The allocation decision is based on data obtained from all transactions.

In an OODBMS context, a third class of allocation algorithms should also be considered. Whereas the terms “local” and “global” refer to transactions (processes) as owners of partitions, the database buffer could also be divided into parts containing a single type of page only. In our example, the buffer could have three partitions, for FPA, DBTT, and USER pages respectively. Again, partition sizes could be static or dynamic. A complete classification scheme is given below as the given classification scheme seems to reflect all buffer allocation algorithms that promise r (successful application and are feasible with a reasonable amount of overhead. The main disadvantage of static allocation (whether transaction oriented or page-type oriented) is its inflexibility in situations where the OODBMS load changes frequently. Since the number of buffer frames allocated to a single transaction remains constant, static allocation is especially inefficient in an interactive environment where transactions can be
blocked by long user think times. Because of its inflexibility, static allocation is not considered to be applicable in database buffer management.

After considered all these issues I tried to address the problem of buffer management in a OODBMS when the workload consists of transactions of different priority levels. I represented a high-priority cards methods as a new buffer management algorithm that uses hints provided by the OODBMS access methods. The performance of Priority-Hints is compared to that of priority buffer management shames introduced earlier for a variety of workloads. My simulation results indicate that Priority-Hints performs consistently better than simple LRU-based algorithms. Furthermore, my algorithm approaches (and in some cases surpasses) the performance of highly sophisticated algorithms that require much more information to be provided to the buffer manager.

1.2 Goals of Thesis

As we all are facing the different issues of buffer management due to increasing demand of immediate response in every data field, Priority scheduling has recently become an area of increased interest to the database community [SIGM88, Abbo88, Abbo89,Care89, Hari90]. Applications that require different levels of system response for different transactions (for example, a system that is designed to provide faster service to interactive jobs than to batch jobs) can benefit from priority scheduling at the OODBMS resources. Several data-intensive application the importance of using priority-based buffer management in a DBMS that already uses priority at the CPUs and the disks may itself be open to question. In my work I investigate all of these issues and their optimal solutions. Secondly my aim is to study the importance of integrated buffer management in OODBMS and its find out the various techniques to improve it and will try to increase the performance and cut the cost of fetching big data and queries.

1.3 Literature review and Background

Integrated buffer management is a very interesting issue and so many teams worked on i.e. want to show some of the related work on buffer management by some famous scholars as follows:
1.3.1 An Evaluation of Buffer Management Strategies for Relational Database Systems

Author: Hong-Tai Chou” David J. Dewitt

Place : Computer Sciences Department University of Wisconsin


Abstract: We describe the second year of work on an integrated system for intelligent compression and transmission of copious data acquired by space borne instruments. At its core, our system contains a wavelet-based progressive image compression algorithm, ICER. Our modified version, ROI-ICER, accepts input priorities measuring the relative importance of various “regions of interest” in the source data, and tags its output packets to reject both the regional priorities and the wavelet bit layer priorities. The output of the data compression module is supervised by an intelligent buffer manager that receives prioritized packets from many different source images and tries to select packets for transmission that will maximize the total science value of the data received on the ground. Our baseline buffer manager uses a simple form of double-valued prioritization: admissions and discards are determined by priorities established by ROI-ICER, while transmissions are first-in, first out (FIFO) among packets that survive the admission/discard process during their time of residency in the buffer.

Extensions of the baseline classification and prioritization algorithms now cover more realistic earth science scenarios, including applications with multispectral data. Extensions to ROI-ICER incorporate a new data model and compression engine. Improvements to the buffer manager handle dynamically changing priorities. A buffer state parameter can be fed back to save ROI-ICER from performing unnecessary computations. Various theoretical advances are being evaluated for inclusion in the mainline software: algorithms to optimize quantizes for feature compression and classification for prioritization feedback, based on a criterion trading off rate, distortion and complexity; improvements to the Mallet distortion model that yield better analytical model-based bit allocations for optimizing region-of-interest coding; and a new buffer control criterion that can approximately match both the minimum worst-case distortion achieved by a minimal criterion and the minimum average squared distortion achieved by a minimum mean squared error criterion.
1.3.2 Title: Buffer Management

Author: Bill McClelland, Partner, AGI

Project: Buffer Calculations

Work: At the start of a project, required buffer sizes are determined according to a Rule of Thumb using the Square Root of the Sum of Squares as a safety net. The underlying principle is as follows: Once work behaviors and incentive systems have been adjusted to eliminate the wasting of safety and once the reporting of early finishes becomes a reality, organizations can count on a canceling effect between tasks that take shorter and tasks that take longer than their estimated ABP times. This means that less safety is required to protect the overall project than would otherwise be required to protect each individual task. The Central Limit Theorem states that the sum of a large number of independent random variables with identical distributions will have a Normal Distribution. This means that 97.5% of the distribution will lie below 2 Standard Deviations (2s) above the mean value. In a perfect world, where all necessary Central Limit Theorem assumptions hold true, the approximate amount of safety required to protect a Critical Chain of length $CC = \sum i$ ABP I would be somewhere in the neighborhood of 2s, where $2s= 2 \sum i \left( (\text{HP }i - \text{ABP }i )/2 \right)$ which simplifies to $2s= \sum i \left( \text{HP }i - \text{ABP }i \right)$ otherwise known as the Square Root of the Sum of Squares. However, reality shows that the assumptions required by the Central Limit Theorem do not hold true.

1. The underlying distribution of each task is different.

2. Tasks are not truly independent.

3. We cannot count on the instant availability of resources to begin work as soon as a task is ready.

These deviations from the required assumptions must be taken into account in

Sizing the Project Buffer. For long chains of tasks, the Rule of Thumb (50% of the safety removed) calculation leads to larger buffers than the S.R.S.S. and has a proven track record for providing adequate protection under these conditions. However, for short chains or, for chains characterized by extreme differences in task variability from one task to another, the Rule of Thumb breaks down. Depending upon the Covariance of the tasks in question, the R.O.T. actually indicates a need for less protection than use of the S.R.S.S. method. The R.O.T. calculation will always be smaller than the S.R.S.S. calculation for chains of 4 or fewer tasks.
If the Covariance is large, this crossover point can even occur at 14 or more tasks. When the S.R.S.S. method calculates a larger buffer requirement than the R.O.T., it is likely to mean that the R.O.T. no longer provides adequate protection. However, the underlying assumptions that led us to the R.O.T. approach haven’t changed either. Thus, it is unlikely that the R.S.S. calculation will provide adequate protection. Not only are the Central Limit Theorem assumptions not valid but, at this point, the Critical Chain itself is short enough that we need to question whether there will be sufficient “aggregation” or “canceling” effect among tasks that exceed their estimated ABP times and tasks that take less than their estimated ABP times. When the R.O.T. calculation is lower than the S.R.S.S. calculation, informed judgment should be used to calculate Project Buffer sizes. That is, a careful inspection needs to be made of the task work in question to determine the likelihood that the associated buffer will provide adequate protection. Some projects have only minor variability at the end; others conclude with highly variable tasks; while still others depend upon the simultaneous convergence of several highly variable pathways (where it feels like winning depends upon back to back to back homeruns). Under these circumstances, consider the S.R.S.S. to be the minimum protection needed. Depending upon the task work under consideration, it might be advisable to increase the Project Buffer size above the S.R.S.S., perhaps up to the total amount of safety removed; \( \sum (\text{HP}_i - \text{ABP}_i) \) or even beyond that amount for extreme situations. The insights of the project manager and project team are essential in this determination, as no absolute mathematical answer is possible. Also note that the above discussion excludes considerations where a Project Buffer needs to be increased to account for anticipated project risk. This entirely valid need is something for consideration over and above any of the Buffer sizing mathematics.

1.3.3 Title: CONSTRAINT MODELING AND BUFFER MANAGEMENT WITH INTEGRATED PRODUCTION SCHEDULER

Author: Li Jun Shen, David, K. H. Chua

Work:

Constraint modeling is a necessary step in construction planning. The basic CPM approach provides a simple and practical means for resolving time-related precedence constraints between activities. However, most CPM-based tools do not support dealing with the constraints regarding resource and information availabilities at the production level planning phase. When
these constraints are concealed in the work plan, it is difficult to assure that they are removed in time so that work takes place as planned.

Consequently, the reliability of work plans/assignments will be reduced. This section presents a scheduling tool called integrated production scheduler (IPS) to handle the no precedence constraints in supply chain and information flow. The IPS has three main objectives to be fulfilled. The first is to promote work plan reliability. The second is to increase resource utilization and throughput based on the estimated resource profile. The third is to maintain a stable work flow through reducing uncertainties in the supply chain and information flow. To further facilitate reliable planning, a set of schedule buffers are established to help manage the constraints. Specifically, the working buffer and the shielding buffer ensure quality assignments by removing resource conflicts and supply chain uncertainties. The pulling buffer and the screening buffer increase resource and information availabilities by managing the delivery issues in advance. With the proposed schedule buffer management, it is feasible to enhance the reliability of look-ahead plans a Constraint management is the core issue in production planning and control during the course of project. It involves identifying the constraints (e.g. manpower, materials, machines, and information prerequisites) and eliminating them to achieve fewer disruptions in production processes and higher throughput against limited available resources. The CPM/PERT-based network scheduling approaches have been extensively employed to identify prerequisite activities and determine activity start/finish times.

Although simple to use and capable of solving sophisticated problems, the basic CPM/PERT methodologies have been criticized for lacking the ability of dealing with non-precedence constraints, e.g. resource constraints. These constraints generally account for the resource and information availabilities that are crucial in determining the actual activity start times. It is necessary to recognize and resolve such non-precedence constraints through effective constraint modeling and management otherwise production plans will be unreliable. Most available methodologies for resolving the non-precedence constraints, or resource constraints in particular, may fall into one of the following categories, namely heuristic (Davis 1973, Elsayed et al. 1986, hanmuganayagam 1989, and Tsai 1996), optimal (Elmaghraby 1993, Yang et al. 1993, Chan et al. 1996, and Schniederjans et al.1996) and simulation (Andersen 1995, and Chan 1997) approaches. These methodologies help find optimal or near-optimal solutions
on the constrained resource allocation problems, thus overcome the weakness of the basic CPM/PERT approaches. A major difficulty in applying these methodologies, however, is that a predetermined resource profile must be supplied first. The profile contains key resource constraints to be satisfied so that any changes on it may affect the schedule subsequently. For this reason, the validity of the schedule is largely dependent on the reliability of the resource profile which comprises a set of variables, rather than constants, changing from period to period.

If the profile is subjected to high uncertainties from the supply chain and work flow, it is unlikely to obtain ‘optimal’ results based on the above methodologies. As a matter of fact, reliable planning often requires more details on the resource and information delivery issues at the production time. These issues represent the ‘hidden’ constraints regarding resource and information availabilities, which are insignificant in the master schedule, important in the look-ahead plans, and essential when assignments are given. Managing these constraints is related to the management of supply chain and information flow, respectively. If they could be explicitly identified and removed in the look-ahead plans, fewer uncertainties would exist and a stable work flow would be achievable. Consequently, the project manager may focus on optimizing resource utilization, reducing excessive inventories, and increasing throughput.

In order to maintain the simplicity of the CPM while modeling more kinds of constraints in the look-ahead plan, Chua et al. (1999) and Shen et al. (2000) proposed a tool, namely Integrated Production Scheduler (IPS), to manage two types of integrated constraints (i.e. RESOURCE and INFORMATION constraints) in addition to PROCESS constraints in production processes. The IPS contributes to reducing the uncertainties of resource and information deliveries through pull-driven production control and messaging systems. This section further discusses the determination of the adjusted activity start times under the influence of non-precedence constraints. To facilitate reliable look-ahead planning, a set of schedule buffers are established, specifically working, shielding, pulling, and screening buffers. These buffers help achieve quality assignments (Ballard and Howell, 1998) through removing constraints in advance and shielding assignments from most foreseen process uncertainties. Eventually, this would help achieve lean process management consequently achieve lean process management.
1.4 Chapter Description:

In this section I want to mention my thesis work according to the main chapter details as following.

- First chapter gives the glimpse of my research work through overview section and goal of my work.
- Second chapter is describe the main Object Oriented Database Management concepts that are useful in the whole research work. I mentioned the basic difference between OODBMS and other traditional databases. I described the architectural issues according to the client and server layer, network model and data transfer grain.
- Third chapter is related to synergistic buffer management and storage management that further I used in the priority algorithm design. In this section I mention the database model that I will use in my research process.

- Forth chapter has cover the description of integrated system model and various buffer management techniques like LRU and Multiple LRU Latches.
- In fifth chapter I introduced a high priority based buffer management algorithm and its experiment and conclusion. I also mentioned the future scope in this area and research.

1.5 Summery

In the above chapter I tried to explain the overview of importance of efficient buffer management techniques by that we can cut the response time of fetch data from database. Since a physical access to a database page on disk is much more expensive than an access to a database page in the buffer, the main goal of a database buffer manager is the minimization of physical I/O for a given buffer size. This goal has to be accomplished under certain restrictions resulting from the interface between the buffer manager and other OODBMS components.