An Overview of Global Seek Optimizing Real-Time Database Algorithms

SALIM Y. AMDANI, ANUPAMA C. GIRAM

Abstract— **The aim of conventional disk scheduling algorithms is to optimize the disk throughput. To meet time constraints, some conventional real-time algorithms, such as Earliest Deadline First (EDF), can be used to schedule disk I/O requests. However, the relative position of requested data on the disks is ignore by such algorithms. Because of large seek time and rotation latency, their throughput is relatively low. Several hybrid real-time algorithms were proposed, to keep a good tradeoff between optimizing throughput and meeting time constraints. The SCAN-EDF combines the features of SCAN type of seek optimizing algorithm with an Earliest Deadline First (EDF) type of real-time scheduling algorithm. Deadline-Modification-SCAN (DM-SCAN) that suggests the use of maximum-scannable-groups compute the suitable request group for seek-optimizing with guaranteed real-time requirements for rescheduling. Global Seek-optimizing Real-time (GSR) disk scheduling algorithm improve the system performance. A new algorithm based on GSR that is called IGSR (Improved GSR). This proposed method improves throughput and decreases the number of missed deadline. Performance evaluation showed IGSR decreased the number of missed deadlines and increased disk throughput in compare with GSR. TGGSR achieves higher disk throughput as compare to EDF and GSR.**

***Index Terms*— Disk Scheduling Problem, Deadline, DM-SCAN, EDF, GSR, IGSR, TGGSR.**

I. INTRODUCTION

In the information age, internet spreads the information worldwide, and is bulk and changing constantly and dynamic in nature. As our society becomes more integrated with computer technology, information processed for human activities necessitates computing that responds to requests in real-time rather than just with best effort. In fact, Database Management systems have entered the Internet Age. It degrades the system performance if too many users approach for information. The degradation may cause delay and trouble for particular end user in accessing the information.

Accessing information in easy way and within certain time limit, by keeping its freshness, assessing user's requirements and then providing them information in time is important aspect.

Like a conventional database system, a real time database functions as a repository of data, provides efficient storage, and performs retrieval and manipulation of information. However, as a part of a real-time system, whose tasks are associated with time constraints, a real time database has the added burden of ensuring some degree of confidence in meeting the system's timing requirements[1][2].

Real time data base systems combine the concepts from real time systems and conventional database systems. Thus, real time database systems should satisfy both the timing constraints with data integrity and consistency constraints.

Typically, a timing constraint is expressed in the form of a deadline, a certain time in the future by which a transaction needs to be completed. In real-time database systems, the correctness of transaction processing depends not only on maintaining consistency constraints and producing correct results but also on the time at which a transaction is completed. Transactions must be scheduled in such a way that they can be completed before their corresponding deadlines expire. For example, both the update and query operations on the tracking data for a missile must be processed within a given deadline: otherwise, the information provided could be of little value.

Example applications that handle large amounts of data and have stringent timing requirements include telephone switching radar tracking and others. Arbitrage trading, for example, involves trading commodities in different markets at different prices. Since price discrepancies are usually short-lived, automated searching and processing of large amounts of trading information are very desirable. In order to capitalize on the opportunities, buy-sell decisions have to be made promptly, often with a time constraint so that the financial overheads in performing the trade actions are well compensated by the benefit resulting from the trade. As another example, a radar surveillance system detects aircraft "images" or "radar signatures". These images are then matched against a database of known images. The result of such match is used to drive other system actions, for example, in choosing a combat strategy.

The goal of transaction and query processing in real-time databases is to maximize the number of successful transactions in the system.

II. BACKGROUND

A. Disk Scheduling Problem

In a disk-based database system, disk I/O occupies a major portion of transaction execution time. As with CPU scheduling, disk scheduling algorithms that take into account timing constraints can significantly improve the real-time performance. CPU scheduling algorithms, like Earliest Deadline First and Highest Priority First, are attractive candidates but have to be modified before they can be applied to I/O scheduling. The main reason is that disk seeks time, which accounts for a very significant fraction of disk access latency, depends on the disk head movement. The order in which I/O requests are serviced, therefore, has an immense

Manuscript received December 19, 2013.

Prof. S. Y. Amdani, Assoc. Professor and Head, Dept. of CSE B. N. C. O. E, Pusad (India) 9764996786

Miss. Anupama C. Giram M.E. Student, Dept. of CSE, B. N. C. O. E, Pusad (India) 8605664555

impact on the response time and throughput of the I/O subsystem.

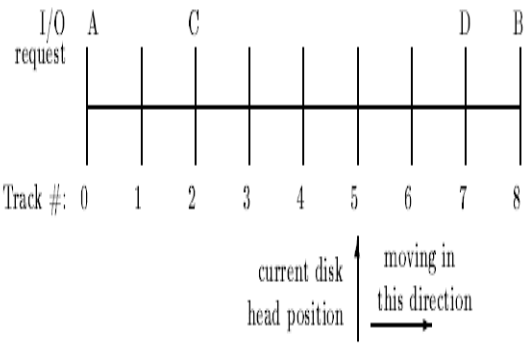

Fig1: Disk Scheduling Example

III. OVERVIEW OF EXISTING ALGORITHMS

A. EDF

In 1973 Liu and Layland suggested the most popular real time disk scheduling algorithm Earliest Deadline First EDF. In EDF transactions are ordered according to deadline and the request with earliest deadline is serviced first.

EDF was originally designed for processor scheduling. When applied to disks, the algorithm simply selects the request with the earliest deadline for service. An advantage of this algorithm is that it is simple and easy to implement. EDF assumed that:

- All tasks are periodic
- Tasks are ready to run at their release time
- Deadlines are equal to periods
- Tasks do not suspend themselves
- Tasks are independent (no shared resources, no synchronization between them)
- No overhead costs for preemption, scheduling or interrupt handling
- Processing is fully preemptable at any point

The idea of EDF algorithm is that the scheduler reevaluates the deadlines of all tasks any time an interrupt occurs (a new task is activated or a task is finished), and allocate the resource to the task having the earliest deadline. The EDF algorithm is good when the system is lightly loaded, but it degenerates as soon as load increases. Critical task may not get priority over non-critical tasks because the closeness of deadline is only deciding factor. EDF fails to work when deadlines of two transactions come same.

B. DM-SCAN

To improve the disk-seek time, SCAN-EDF algorithm was proposed by Reddy and Wylie in 1993. The SCAN-EDF disk scheduling algorithm combines seek optimization techniques and EDF in the following way. Requests with earliest deadline are served first. But, if several requests have the same deadline, these requests are served by their track locations on the disk or by using a seek optimization scheduling algorithm for these requests. But the efficiency of this algorithm relies on the number of requests that have the same deadline.

To overcome this problem, in 1998[4] R. 1.Chang, W K. Shih, and R. C. Chang proposed Deadline- Modification-SCAN (DM-SCAN) that suggests the use of maximum-scannable groups compute the suitable request group for seek-optimizing with guaranteed real-time requirements for rescheduling. The DM-SCAN identifies and reschedule the maximumscanable- group repeatedly. In this algorithm, request deadlines are reduced several times during the process of rescheduling to preserve EDF schedule.

MSG concept is based on an EDF input schedule. However, in real-world systems, the problem input may not be an EDF schedule. To resolve this drawback, a DMS (*deadline-modification-scan*) algorithm is proposed to modify the original problem input and transfer it into an EDF input schedule. For example, given the schedule sequence *Ti Tj,* and then simply modify the deadline *di* = min(d*i, dj)* to satisfy the EDF requirements $di \leq dj$. This O*(n)* procedure is called *deadline-modification (DM)*. After executing the DM algorithm, it can be proved that the new problem input is an EDF schedule. The MSG concept can be applied to this new problem input to further improve the disk throughput. Notice that, the rescheduled result of MSG may violate the original EDF property. The DM algorithm is again executed to produce a new EDF problem input. Thus, the MSG concept can be applied repeatedly to further improve the disk throughput. There are some limitations of DM-Scan. The input Should be necessarily to satisfy EDF schedule[5].Groups formed are narrow sized and deadlines formed are fictitious deadlines. It is a Local group seek optimization algorithm[6].

C. RG-SCAN

Unlike DM-SCAN, Reschedulable-group-SCAN (RGSCAN) suggested by H. P. Chang, R. I. Chang, W. K. Shih, and R. C. Chang in 2002[5], does not require its input disk requests to be sorted by their deadlines. It also forms larger groups without any deadline modification.

There are two conditions for making the R-Groups. The conditions are as follows:

1. $fi+m <= \min \{d_i\}$

2. $\text{Max } \{r_i\} <= s_i$

test transaction, d_i is the deadline of current transactions, r_i is the release time of current transactions and s_i is the start time of current transaction in a group.

The above conditions can be given in terms of words as follows:

1. The finish time of current transaction should be less than the minimum deadline of group if there exist a group or compared with the deadline of previous transaction.
2. Maximum release time of group is compared with the start time of current transaction and it should be less than the start time of current transaction.

If both these conditions are satisfied then the group of these transactions is formed otherwise next transaction is taken into consideration.

After making these R-Groups, it select only the mutually exclusive R-Groups means the repeated groups are neglected, then after selecting these mutually exclusive R-Groups the SCAN algorithm is applied on that groups. In RG-Scan also the seek optimization is within local groups and formed groups have narrow sized[6].

D. GSR

In SCAN-EDF, DM-SCAN and RG-SCAN algorithms rescheduling is only possible within a local group of requests. H.-P. Chang in 2007 [6], suggests Global Seek-optimizing Realtime (GSR) disk scheduling algorithm that groups the EDF input tasks based on their scan direction. These tasks are moved to their suitable groups to improve the system performance in terms of increased disk throughput and decreased number of missed deadlines. GSR schedules are always feasible if the input real-time disk requests are EDF feasible sequence. But with an infeasible input, it is very unlikely to have a feasible output. This is due to the fact that after each regrouping of input tasks, GSR checks the feasibility of the new schedule. If the new schedule is infeasible, GSR algorithm ignores the movement and selects another request to regroup and this continues until it reaches the last request

Three Main Steps of GSR :

- EDF to SCAN mapping
- Scan Group Identification(SGI)
- GSR algorithm

Although SCAN can maximize data throughput, its schedule result does not meet the timing constraints of real-time transactions. In contrast, the EDF schedule is good for real-time requirements. However, its disk throughput is low. EDF-to-SCAN mapping is a bipartite mapping obtained by connecting each node in EDF schedule to corresponding node in SCAN schedule with an edge. After identifying all scan-groups, GSR tries to reschedule each task into all scan-groups before its own scan-group and to compute the improvement of data throughput of each rescheduling result. The one with the largest throughput improvement while guaranteeing a feasible schedule is selected for rescheduling (the task is rescheduled from its own scan-group to the new one). In GSR Complexity is more due to the number of internal calculations and overhead is more[7].

E. IGSR

In 2009 Hossein Rahmani, Mohsen Ebrahimi Moghaddam proposed IGSR[7]. This method improves throughput and decreases the number of missed deadline. When the input is infeasible, IGSR provided more feasible output Schedule. It accepts a sequence of tasks that are ordered by their deadline. MSGI is basedon Scan Group Indentification Algorithm[6].MSGR works same as GSR but instead of rescheduling whole input it reschedule the selected tasks in best scan fit scan gruup.

IGSR uses threen main subroutines that are

- FTS Finder
- MSGI (Modified Scan groups Identification)
- MGSR (Modified GSR)

The input sequence is ordered by their deadline. IGSR calls FTS Finder (Algorithm) iteratively on the input sequence t. In each iteration, the found FTS is used as the input sequence to MGSR subroutine. Consider task Ti as the first task that misses its deadline and it is returned by FTS Finder. With regards to new order of tasks in the rescheduled FTS by MGSR and the probable improvement in total fulfill time; Ti has more chance that it does not miss its deadline and be added to the new FTS. If Ti is not added to this FTS, it is given a second chance by calling MGSR to reschedule Ti into the FTS. If the output of MGSR is still infeasible the task Ti is ignored completely and the next task in the input schedule t will be checked. The next task is selected on the basis of First in First Served (FIFS) policy. IGSR algorithm do not account for the track distance between the last transaction of current scan-group and the first transaction of the next scan-group, which will decrease data throughput of real-time system.

F. TGGSR

Nianmin YAO, Jinzhong CHEN, Ang LI in 2012 proposed TGGSR[8]. This approach first generates the initial schedule considering the track locations of the transactions and then this input schedule is decomposed into the SCAN groups and final rescheduled result is found. In TGGSR instead of taking input schedule in EDF it assign the priorities using **$Pi= \alpha*Di + (1- \alpha)*Ai$** where, ($0< \alpha < 1$) is the percent of the deadline Di for Ti, Di is the deadline of Ti, and Ai is the track location(Start Block) of Ti. The transaction having low Pi value has high priority. Then mapping is done in between optained schedule and Scan Schedule. And then second step is applied i.e. Scan Group Identification to obtained Scan Groups. And at last GSR is applied to find final schedule. TGGSR achieves higher disk throughput as compare to EDF and GSR.

IV. CONCLUSION

The real-time disk scheduling algorithms lowers the disk efficiency as they process disk I/O request in order of deadline. To solve this problem, disk efficiency was improved with scheduling algorithms using the SEEK OPTIMIZATION technique. For this group of consecutive transactions are made on the basis of certain condition and then SCAN algorithm is applied within these groups locally or globally. The gap between the performance of processors and the performance of disks is enlarged and the performance bottleneck in a computer system is shifted to the storage subsystem. When multimedia applications become more important, the gap problem becomes more serious. This motivates the need for more efficient use of the transaction scheduler in order to maximize disk throughput and minimize seek time.

In this paper we have discussed overview of various local and global seek optimizing real time database algorithms, which includes DM-Scan, RG-Scan, GSR, IGSR,TGGSR.

REFERENCES

[1] G Ben Kao and Hector Garcia-Molina 1993 "An Overview of Real Time Database Systems", in proceedings of NATO Advanced Study Institute on Real-Time Computing St. Maarten, Neatherland Antilles, Springer-Verlag,

[2] Jayant R. Haritsa Miron Livny Michael J. Carey" Earliest Deadline Scheduling for Real-Time Database Systems"Computer Sciences Department University of Wisconsin Madison, WI 53706

[3] R. Abbot and H. Garcia-Molina. 1988 "Scheduling Real-Time Transactions: A Performance Evaluation",Proceedings of the 14th VLDB Conference ,Los Angeles,California.

[4] Chang, R.I., Shih, W.K., Chang, R.C., 1998. "Deadline-modification-scan with maximum scannable-groups for multimedia real-time disk scheduling." In: Proceedings of the 19th IEEE Real-Time Systems Symposium, pp. 40–49.

[5] Chang, H.P., Chang, R.I., Shih, W.K., Chang, R.C., 2002. "Reschedulable-Group-SCAN Scheme for Mixed Real-Time/Non-Real-Time Disk Scheduling in a Multimedia System." J. Syst. Software 59 (2), 143–152

[6] Hsung-Pin Chang , Ray-I Chang , Wei-Kuan Shih , Ruei-Chuan Chang, 2007. "GSR: A global seek-optimizing real-time disk-scheduling algorithm." The Journal of Systems and Software 80 198–215.

[7] Hossein Rahmani, Mohsen Ebrahimi Moghaddam 2009 "An Eficient Disk Scheduling Algorithm for Multimedia System Based on GSR Algorithm" IEEE International Advance Computing Conference (IACC 2009) Patiala, India, 6-7 March

[8] Nianmin YAO, Jinzhong CHEN, Ang LI, 2012 "An Inter-group Seek-optimizing Disk Scheduling Algorithm for Real-time System" Journal of Computational Information Systems 8: 18 7579-7586

Prof. S. Y. Amdani received his M.E. CSE degree from SGB Amravati university, Amravati in 2008, and research scholar from 2009. Working as Associate Professor in Deptt. Of CSE B.N.C.O.E., Pusad (India), & life member of Indian Society for Technical Education New Delhi

Miss Anupama C. Giram received her B.E. CSE degree from SGB Amravati university, Amravati in 2012. M.E. Appered. Working as Assistant Professor in DEptt Of CSE B. N. C. O. E. ,Pusad (India).