Data Distribution and Improving Disk Performance for Faster Memory Access

Shihabudheen P M, *Member, IACSIT*

***Abstract*—The layout of traditional disks is optimized for file systems with large sequential access. For these sequential workloads, cost of seeks and rotational latencies are amortized on the transfer of a large amount of data to/ from the disk. Each access from a file system workloads, however incurs a time consuming seek and rotational latency that dominate the request response time, resulting in very less data rate. In the designed model to enhance the speed of data accessing from the hard disk and there by improving the over all performance of the system, we distribute one file in to the different platters available in the hard disk. For this, first split the file in to the number of parts based on the block size and the number of platters available in the hard disk by the file splitting storage algorithm. The first part is stored on the first storage surface. Then, the second part is stored on the same cylindrical location of the second storage surface. This process is repeated for the remaining parts of the file. During the reading process, place the head dedicated to the storage surface to the correct location. Then perform the reading simultaneously. The access time can be reduced drastically. The main intention behind the designs model is to supply the data as fast as possible to the RAM. This model is helpful for enhancing the virtual memory and paging mechanism by creating an illusion of the presence of data in the RAM, so that user can execute his program more efficiently with less amount of RAM.**

Index Terms—Hard disk drive, internal data transfer, internal fragmentation, response time.

I. Introduction

Page oriented work loads are commonplace in many real world applications varying from virtual memory subsystem to database buffer pool managements. This work loads are characterized by small fixed size, random access reads and writes [1]-[3]. The layout of traditional disks is optimized for workloads with large sequential accesses. For this sequential workloads, seek and rotational latencies are optimized over the transfer of a large amount of data to/from the disk and the application sees good data transfer rates. Each access from a page oriented workload, however incurs a time consuming seek and rotational latency that dominate the request response time, resulting in very low data rate. Each access from a page oriented workload, however incurs a time consuming seek and rotational latency that dominate the request response time, resulting in very low data rate. Today the data access from the hard disk is performed in the serial manner [4].i.e. one location of the hard disk is reading at a time. In many of the operating system the memory allocated is fixed block. It leads to the internal fragmentation. The randomly allocated space leads to the external fragmentation. In fact, we can't utilize the full amount of memory available in the hard disk. The most important drawback of the existing system is of large access time. After the read head starts to read the location, the disk will waits until a block of data is read completely. Only after completing one block it sends data to the RAM. Then the disk drive is needed to wait to get the next block and so on. This will affect the overall performance of the system since the virtual memory concept influences the execution speed. In the designed system, we enhance the speed of data accessing from the hard disk and there by improving the over all performance of the system by distributing one file on to different platters available in the hard disk.

Manuscript Manuscript received February 17, 2012; revised March 30, 2012. This work was technically and financially supported in part by SCMS school of engineering and technology, Cochin, India.

Mr. Shihabudheen P M was with the department of computer science and engineering, SCMS school of engineering and technology, Cochin, India. He is now with the Broadcast Business Unit, Tata Elxsi Limited, Trivandrum, India (e-mail: shihabudheen27@gmail.com).

II. Related Works

Numerous researches are concerned with the problem of reducing the cost of read and write of the disk subsystem. Typically the technique used involves some combination of data replication (mirroring), striping across multiple disks, dynamic data placements and track alignment. Disk shadowing [5] is a technique that has long been used to improve the disk performance, redundancy and availability. Each disk serves as an identical copy of the others. Writes are copied on to each disk, and reads can be serviced from any of the disks. Since only the first write must make it to stable storage to ensure consistency, the disk with the smallest access latency cost determine the performance of writes. Likewise, reads can also be serviced by the disk with the smallest latency. Hou and Patt [6] and Dishon Liu [7] demonstrate policies, which copy data on to many disks to improve performance under simulation and examine the benefit of storing multiple copies of a file throughout a file system spanning multiple storage nodes. S.W.mg [8] examines analytically the benefit of mirroring data within a single disk. In particular blocks are placed twice, on the disk, 180 degree out of phase of one another in the same track. The result is that the average rotational delay is reduced by a factor of two, improving read performance [9]. Also mirroring blocks on to different tracks to help reduce seek latencies.

III. Dynamic Data Placements

There are thousands of file systems for file management. All of them store the data in the hard disk in similar manner.

Only the management scheme may differ. In hard disk the space is allocated in blocks. It suffers from internal fragmentation. Random memory allocation leads to external fragmentation. In the proposed system, we distribute one file on to the different platters, available in the hard disk. For this, first split the file in to the number of parts based on the size and the number of the platters available in the hard disk by the file splitting storage algorithm. The first part is stored in the first storage surface. Then, the second part is stored on the same cylindrical location of the second storage surface. The process is repeated for remaining parts of the file. Dynamic data placement characterizes system where the mapping of logical blocks to physical blocks changes, usually allowing the system to easily write data to the closest free block when a write request is made.

IV. Track Alignment

Y. dishon [4] uses two disks, a logging disk and a data disk, to improve synchronous write performance. During reading, the head is moved a large distance and the spindle motor operation is more. The accessing speed is reduced and the power consumption is more due to the spindle motor rotation. The disk head of the long disk is kept over relatively free track, and log writes can be issued without seek and minimal rotational latency. In our designed system during the reading process place the heads dedicated to the storage surface and place it to the correct location. Then perform the reading simultaneously. So the access time can be reduced. Number of storage space available in the hard disk is two times the number of platters available. Here the storage space is more compacted. That is the allocated space for a file is not spread in the hard disk. So the rotation needed by the spindle motor to read a particular file is reduced. It will help to reduce the power consumption and there by the performance of the system.

V. Hard Disk Drive Specifications

A. Head Switch Time

In the hard disk drive, only one head is active at a time. After reading the first block from one platter surface, if the second data block is on the other platter, the head corresponding to that platter surface must be activated. It is a pure mechanical operation and it takes considerable amount of time which is known as the head switch time. Heads in the HDD is mounted on a head actuator which is responsible for the movement of the head in radial direction. During the head switch time, the actuator is static.

In the designed system, all heads corresponding to each storage surface is active during reading and writing of data. So there is no head switch time.

B. Internal Data Transfer Rate

All hard drive data transfer includes two separate storage locations- the hard drive buffer and system memory. The internal data transfer rate is the amount of data transfer from hard disk platter to the hard disk buffer in unit time. The external data transfer rate is the amount of data transferred from hard drive buffer to the system memory in unit time. In a HDD, the internal data transfer rate is always less than the external data transfer rate (since the HDD is purely a magnetic device and it takes considerable amount of time to read or to write the data).

In the proposed system, we try to increase the internal data transfer rate. Once the data is available in the hard disk buffer, it can be send to the RAM with the already existing SATA or PATA method.

VI. Designed Scheme and Algorithm

In the existing system, the data is stored in cylindrical positions to reduce the head movement. It writes the data in one track completely, and then writes on the same track of the second surface. In the existing system, accessing time is more due to head switch time, seek time, and latency. In the designed system, the data is evenly distributed to platters so that, it can read parallel using heads to the different storage surface.

Storage surface= (2*no .of platters)-2.

The access time can be reduced to and a reduction in the rotation of spindle motor is achieved to save energy.

File splitting storage algorithm (int n):-

1) start
2) divide the file into number of parts each of having size n
3) if the number of the parts is greater than the number of the storage surfaces then repeat the step from 3 to 13
4) put the file blocks which is same as the number of storage surfaces available in the disk into the hard disk buffer
5) if name of the file is not existed in the file allocation table, make entry in the file allocation table
6) else make the modification in the file allocation table
7) write the data from the buffer to the to the cylindrically similar position of the different storage surfaces
8) then the number of file parts remaining is number of the file parts-number of the storage surfaces
9) if the new number of file blocks of size n is less than the number of the available storage surfaces, then repeat the step from 9 to 13
10) if the value of n is 16 bit, then divide remaining file portions into number of blocks having length 16 bits and store it into the cylindrically similar positions of different storage surfaces and go to step 14
11) file size becomes the file size-the stored data size
12) n = n / 2
13) divide the remaining file portion into the number of blocks each of having length n
14) stop

In the algorithm n is the maximum block size supported by the file system. During reading also this algorithm is used to control the head movement. In the designed system, the bits are present in similar cylindrical position of the different platters.

VII. Implementation

The core of the system is a paged disk drive. This disk

drive allocates and manages blocks to maximize random read and write performance of page sized block of data. Install a hard disk with specific properties. In Linux, file system can be implemented more easily. The fuse module provides standard method to control file operation. User applications such as data base and virtual memory systems interface with the paged file system, which provides a page-based instead of file-based view of storage. It is possible that performance gain will be realized by simplifying the file system. But the file system of our investigation improves the ability of a disk drive to minimize rotational and seek latency.

There are three implementation methods to enhance the hard disk performance and thereby the system as a whole.

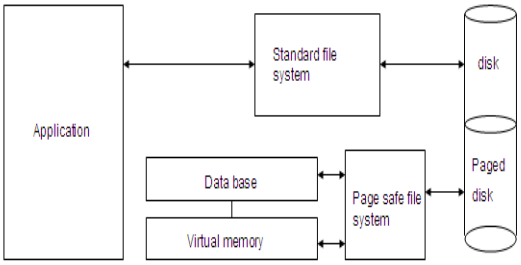

Fig. 1 .System model

A. Block Splitting Storage Method

Most of the operating system uses the fixed blocks of size 1KB, 2KB or 4KB. While storing the block into the hard disk split the block into the number of parts as same as the storage surface available in the disk. Then, distribute these parts to the platters. I.e. store the different parts into the cylindrically similar positions of different storage surfaces. So when reading one block, it can read in parallel manner and send to the RAM. In this case we can preserve the existing file systems with its normal operations. The modification is done in the hard disk drivers only.

B. File splitting Storage Method

In this implementation scheme, first we divide the entire file according to the file splitting storage algorithm. The input to the algorithm is the maximum block size. The maximum block size can be selected according to the file system. Then distribute the blocks to the different storage surfaces as the algorithm is proceeds. In buffer, separate space should be allocated for data from/to each platter.

During the reading process, the first half of buffer space is allocated for all the platters. After completing the first read operation, the first half of the buffer contains the data. This data, we can't send simultaneously to the RAM because of the smaller size of interface cable. During this process, the hard disk should start to read the second portion of the file and put the data into the second half of the buffer. Similarly the buffer allocation is switched between these two portions to avoid the missing of data.

C. File and Block Splitting Storage Method

First, take one block size of the data from the file. Then, split it into the number of parts and store as mentioned in the fist case. Then the remaining portion of file is split and store as mentioned in the second case. So the response time can be reduced along with faster access and fragmentation removal.

To implement the above described methods, first we need a hard disk drive with specified properties. In the present scenario, the heads of the hard disk is mounted on to single arm and head switching is performed to read data from different platter surfaces. In the new system, all the heads should be active at a time to read the data from different platters simultaneously. To implement the second and third methods we have to modify the file allocation table, since most of the operating systems use the fixed block size. But in the designed system, we use blocks of variable size.

VIII. Results

It is often difficult to evaluate the performance of modifications to a disk, as the result depends on many factors, ranging from workload parameters to the design of the disk hardware.

The existing hard disk drive technology is flexible and efficient to perform the storage management, but it lags in data transfer rate and utilization of the storage area. This leads to the degradation of system performance as a whole.

In the graph x axis represent the time in microsecond and y axis represent internal data transfer rate in MB. The horizontal lines indicate the time required to place the head on the starting place of the block. Assume that 2 platters are available in the disk. So there are 4 storage surfaces in the disk and the maximum block size is 4 KB.

In the block splitting storage method, distribute a single block onto different platter surfaces, so that each platter surface contains 1KB of data and they are at the cylindrically similar positions. When the data is read simultaneously from all these surfaces, the time taken to read a block becomes $1/4^{th}$ of the time taken in existing method. The seek time is same in both the cases. Due to the reduction in the rotation of spindle motor the power consumption also gets reduced.

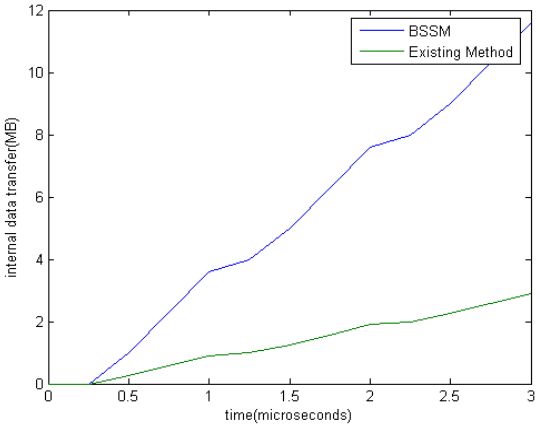

Fig. 2. Block splitting storage method (BSSM)

In file splitting storage method, the entire file is distributed to different storage surfaces in the disk. The maximum block size is 4KB. First store the 4*4KB on 4 storage surfaces. So the 16 KB of data is stored in to the disk. While reading, this 16KB of data can read within the time taken to read one block of data in the existing system. So the internal data transfer rate is 4 times as in the existing system. If the remaining file size is not sufficient to split to 4, 4KB blocks then the block size is reduced to 2KB. This procedure is repeated in the entire life cycle. So the internal fragmentation can be reduced to a great extend.

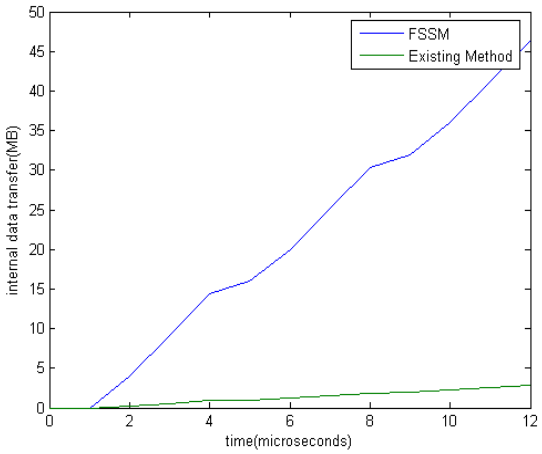

Fig. 3. File splitting storage method (FSSM)

In the block and file splitting storage method, the above two methods are combined. The second method improves the internal data transfer rate to a great extend. But the response time is same as the existing system. In this first 4KB of file is extracted and split into 4, 1KB blocks and stored in different storage surface. The remaining file data is stored according to the file splitting storage algorithm. So the response time is almost 1/4th as that of existing system and the internal data transfer rate is 4 times as that of exiting system.

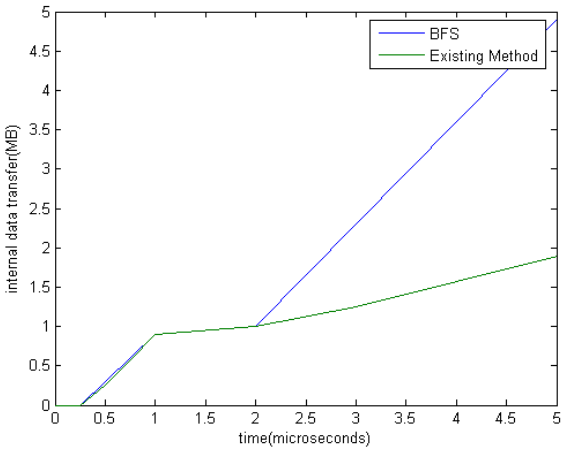

Fig. 4. Block and file splitting storage method (BFS)

The table below analyzes the various optimizing parameters for the three proposed methods. 'X' is a variable which represents the value of a particular parameter in the present hard disk system. later, the value of same parameter in the present system is mentioned in terms of 'X'.

TABLE I: PARAMETER COMPARISON

Parameter	Existing system	Block Splitting Storage Method	File Splitting Storage Method	File and Block Splitting Storage
Internal data transfer rate	X	4X	4X	4X
Response time	X micro sec.	X/4 micro sec.	X micro sec.	X/4 micro sec.
Internal fragmentation	Normal	Normal	Reduced	Reduced
Spindle motor rotation	Normal	Reduced	Reduced	Reduced

In short, by the file and block splitting storage method, the hard disk storage mechanism can be optimized in terms of internal data transfer rate, internal fragmentation, response time and power consumption of spindle motor.

IX. CONCLUSION

The main intention behind this technique is to supply the data instantaneously to the RAM. It will improve the execution efficiency of the system. The designed system enhances the features of virtual memory. The paging mechanism is enhanced to create the illusion of the presence of data in the RAM, so that user can execute his programs more efficiently with less amount of RAM. The internal fragmentation in the system is drastically reduced. This is possible by effectively splitting the file and allocating the space in accordance with the space of each part.

REFERENCES

[1] Diana Bitton and Jim Gray, Disk storage shadowing in proceedings of the fourteenth *international conference very large data bases*, pp. 331-338, 1988.

[2] C. Chao, R' English, D. Jacobson, A. Stepanov, and J. Wilkes, "Mime a high performance parallel storage device with strong recovery guarantees." 1992.

[3] Jzi cker chiweh and lan Huang, "Track based disk logging." In *international conference on dependable system and networks* (DSM '02), pp. 429-438, 2002.

[4] Y. Dishon and T. S. Lui, Disk dual copy methods and their performance in providing of *18th international symposium on Fault tolerance computing (FTCS-18)*, pp. 314-318, 1988.

[5] R. M. English and A. A. Stepanov, "Loge; A self organizing storage device." In proceedings of *USENIX winter g2 technical conference*, pp. 237-251, USENIX, January 1992.

[6] R. Hou and Y. N. Paft, Trading disk capacity for performance in proceedings of the *2nd international symposium on high performance distributed computing*, pp. 263-270, 1993.

[7] Christopher lumb, Jiri schindler, Gregores R. Ganger, Erile Riedel, and David Nagle, "Towards higher disk head utilization extracting free band width from busy disk drives." In proceedings of *the 4th symposium on operating system design and implementation [OSDI]* pp. 87-102, October 2000.

[8] S. W. mg, "Improving disk performance via latency reduction." *IEEE transaction on computers*, pp. 307-316, January 1993.

[9] J. schindler, J. Griffin, C. Lumb, and G. ganger, "Track aligned extents; matching access patterns to disk drive characteristics." 2002.

Mr. Shihabudheen P M was born at Kerala, India on 27th March, 1989. He complete his bachelor degree in computer science and engineering from SCMS school of engineering and technology, Cochin, India affiliated to Mahatma Gandhi university, Kerala, India in the year of 2011. He has published number of research articles in the area of computer architecture and image analysis. He is currently working as a engineer in Tata Elxsi Limited, Trivandrum, India. He was nominated as a member of international technical committee for International Conference on Computer Technology and Development held at china in the year of 2011. His research interests includes theoretical computer science and its applications, representation and transmission of data in the domain of digital video and audio broadcasting.

Mr. Shihabudheen was an active member of Computer society of India (CSI). Now is a member of International Association of Computer Science and Information Technology (IACSIT) since 2011.