Mansoura Engineering Journal

Volume 15 | Issue 2 Article 8

5-22-2021

Performance Evaluation of Movable Head Disk Schedulers.

Mofreh Salem
Associate Professor., Computers and Control Engineering Department., Faculty of Engineering., El-Mansoura University., Mansoura., Egypt.

Mansoura Engineering Journal (MEJ) Vol. 15, No. 2, Dec. 1990

PERFORMANCE EVALUATION OF MOVABLE HEAD DISK SCHEDULERS

تقييم أداء طرق ادارة الأقراص ذات الرؤوس المتحركة

MOFREH M. SALEM

Computers [illegible] Control Dept., Faculty of Engineering,

El-Manscura University, El-Mansoura, EGYPT

ملخص البحث..

نظام التشغيل هو عبارة عن مجموعة من البرامج التى تربط جهاز الحاسب الالى بمستخدميه ... والغرض من تصميمة هو الاستغلال الامثل لمكونات الحاسب المختلفة مثل الذاكرة و المعالج و المعلومات و وسائط الادخال و الاخراج ووسائط التخزين ..والاخيرة يمكن تقسيمها من حيث الوصول الى ثلاثة اقسام رئيسية هى : الوصول التتابعى (مثل الشرائط) و الوصول المباشر الكامل (مثل القلوب) و الوصول المباشر (مثل الاسطوانات و الاقراص). . والاقراص يمكن ان تحتوى على رؤوس ثابتة او رؤوس متحركة .

حاليا فان معظم العمليات التى تتم فى الحاسبات الحديثة تستخدم النوع الاخير حيث تمتاز بكفاءة اداء وتكاليف افضل اذا ماقورنت بالوسائط الاخرى ورغم ذلك فان استخدام مثل هذه الاقراص من الممكن ان يؤدى الى نتائج عكسية تقلل من كفاءة الجهاز ككل .. اذا لم تدار بطريقة مناسبة لخدمة المطلوب منها ...لهذا كان اختيارطريقة ادارة القرص يحظى باهمية خاصة فى انظمة تشغيل الحاسبات المختلفة ... ولذلك فان هذا البحث سوف يركز على الطرق المختلفة لادارة الاقراص ذات الرؤوس المتحركة .

وفى هذا المجال ..فلقد قام الباحثون المهتمون بهذه المشكلة [illegible] العديد من الطرق للقيام بهذه المهمة ...ولكن هذه الابحاث [illegible] باقتراح الاعتبار كثير من العوامل التى يمكن أن تؤثر على كفاءة [illegible] لم تاخذ فى المقترحة حيث انه لم يتم اختبارها على نظم حقيقية . [illegible] الاداء للطرق الى كثير من الاستفسارات..منها على سبيل المثال. [illegible] و هذا يؤدى بدورة ومع اى نظام..يمكن تفضيل طريقة على آخرى؟ [illegible] تحت اى شروط وظروف

وبناء عليه فلقد تم التركيز فى هذا البحث على تحليل ثلاثة من الطرق الاكثر انتشارا ثم تم اختبارها عمليا على جهاز الحاسب الخاص بابحاث القسم تحت ظروف تشغيل مختلفة لقياس المعاملات التى تم استخدامها فى تقييم اداء الطرق المختلفة ..حيث خلص البحث الى النتائج المعملية والمقارنات المدونة فى نهايته .

وهذه النتائج ساعدتنا فى اختيار افضل الطرق التى تتناسب مع ظروفنا من حيث الجهاز و الاحمال...وكذلك فانها يمكن ان تساعد المسؤلين عن تشغيل نظم الحاسب المختلفة للاسترفاد بها فى امكانيات تحسين كفاءة اداء ادارة الاقراص لديهم .

ABSTRACT

In recent years, several disk scheduling techniques have been suggested. It seems clear that the choice of n algorithm can make a significant difference in computer system performance, but which is best is still an open question. Therefore, the main contribution of this paper is the detailed computational experience of three well- known scheduers, to evaluate their performances. The aim of this evaluation is to help those responsible for system operations in assessing the possibilities for improving disk performance. Finally, it is to help us to choose the best scheduler suitable for our MVAX system with its load conditions.

1- INTRODUCTION

An operating system is a collection of algorithms which act as interface between the computer system and the users. It is designed to manage the processors, the memory, the devices, and information. The scheduling of these resources allows the system to operate more efficiently. Devices can be generally categorized into two major groups; I/O and storage devices. According to the access technque, there are three main types of storage devices; serial (tape), complete direct access(core),and direct access (drum,disk). The latter may be fixed head disk which has one head for each track, or moving head disk which has one head to access different tracks. It is characterized by small variance in access time, and can be efficiently shared and simultaneously used providing high performance rates. Therefore, most of the processing of modern computer systems prefes to use on the disk system. However, in multiprogrammed computing systems, inefficiency is often caused by improper use of the disk, because its performance is critical factor to the performance of the entire system. Hence the proper management of disk storage is of central importance to a computer system.

One easily tunable piece of a disk system is the disk scheduling algorithm which the operatino uses to decide which of a set of pending disk requests should be served next. An effective system way to improve system performance is to improve this algorithm. Therefore, in the last and low cost disk scheduling algorithms have been suggested [1-12]. Among them we note the years, several and its variants [3 - 9] and recently SVS [5], as improvements over the SSTF [1,4,9], SCAN analysis has appeared for most of these algorithms to indicate how the FCFS. However, no exact system measurements. It is not clear how such algorithms they are actually used on actual compare on real systems, and how the arrival process affects the performance criteria. Also, it is not known under what circumstances an algorithm is better than the other. It seems that these questions will represent formidable analytic problems for some time to come. Consequentaly, this paper is again concerned with the performance evaluation of the most popular disk scheduling techniques. In the next section, the reformulation and the interrelationship of these techniques in terms of different times are discussed. The simulation model and its parameters are presented in the third section. while in the last two sections, series of experimental results are discussed and followed by the conclusion.

2- PROBLEM FORMULATION

The hardware for a disk system can be divided into three parts; controller, drive and disk. The disk controller determines the Logical interaction with the computer. It takes the instructions from the CPU and orders the disk drive to carry out these instruction. The disk drive is the mechanical part, including the device motor, the heads and associated Logic. The disk is a flat circular shape with two surfaces. Each surface is divided into tracks. within a track, information is stored in sectors (or blocks) with fixed or variable size. While, the cylinder means all of the track on the different surfaces on one drive that can be accessed without moving the heads. Thus information on the disk may be referenced by a multi- part address including the drive number, the surface, the track, and the sector. Therefore, to access a particular block of data on the disk, three operations are usually necessary. First, the system must move the head to the appropriate track. seek operation, in a seek time (T_s). The head must wait until the desired block rotates under or over the head . This delay is called latency operation and takes latency (or rotational) time (T_l) . Finally, the actual transfer of data between the disk and main memory takes place .The last part is then transfer time (T_t). Hence, the service time (T) takes the form :

$$T = T_s + T_l + T_t \tag{1}$$

This equation can be rewritten in the following formula :

$$T = T_s + T_r / 2 + T_r / m = T_s + A \tag{2}$$

where T_r = disk rotation time, and m = number of sectors per track.

Expected service time for requests on the same cylinder as the current request is found by first noting that each cylinder has (t) tracks and total of (mt) sectors. If the current request is on sector i, the probability that sector i will be chosen again is(t-1)/(mt-1). The probability that a specific sector,j = i, will be chosen as 1 / (m - 1) 1 - (t - 1) / (mt - 1) ..

The expected number of sectors traversed to service the next requests [1 , 2] :

$$B = \sum_{k=0}^{m-1} K \text{ prob \{ sector K is next \}} + 1 = \sum_{k=0}^{m-2} K \frac{1}{m-1}\left[1 - \frac{t-1}{mt-1}\right] + (m-1)\frac{t-1}{mt-1} + 1$$

$$= \frac{(mt-2)(m-1)}{2(mt-1)} + 1$$

$$T = (Tr / m) \text{ (expected number of sectors traversed)}$$

$$= P(T_s + A) + (1 - P)(T_r / m)(B) \tag{3}$$

where P is the probability that the next request to be serviced is not on the current cylinder, its value is depended on the scheduleing algorithm. If the desired disk drive and the controller are avaible, the disk request can be serviced immediately. Otherwise, as in the case of multiprogramming system with many processes, the disk requests will need to be queued. Furthermore, because the above disk operations involve mechanical movement, the time T is often an appreciable fraction of a second. This is very slow compared with the very high processing speeds of the central computer system. So, it is importrant that the operating system must improve the average disk service time by scheduling the requests and hence move the head to the desired track as fast as possible, i.e. the requests must served with minimum time. Consequently, disk scheduling must involve a careful examination for the positional relationships among the waiting requests to determine the most efficient way to service the requests. The common schemes of disk scheduling are latency optimization and seek optimization. It seems that the minimizing latency time usually has very little effect on overall system performance, except under very heavy loads. Also, seek times tend to be about an order of magnitude greater than latency time. So that, most disk scheduling algorithms concentrate on minimizing seek times. Since scheduling algorithm can minimize time wasted in performing lengthly seeks, the throughput (number of requests served per unit time) can certainly be maximized and can then be improved. In the same direction, the disk scheduler should attempt to minimize the average waiting (the mean response time).Because, the expected waiting time, T_w , for individual requests from time of arrival into the queue until completion of service is equal to L / λ , and the disk utilization ρ is of λ T. Where L is the mean queue length includes request currently being served, and I is the input rate in requests per time unit. lso, the scheduler must minimize the variance in response times. It is defined as a measure of how far individual items tend to deviate from the average of the items, and is used to indicate predicatability. It means the smaller the variance, the greater the predictability . Several disk scheduling algorithms have been proposed to deal with the seek minimization [1-12]. Some of the most popular algorithms [1,3,4,9] will be reformulated and discussed in detail as in the following.

2-1 FCFS Algorithm

The simplest form of disk scheduling is the first-come-first-served (FCFS). It does not take advantage of positional relationships between I/O requests in the current queue or of the current position of the head. The requests are queued and linked list, new requests are added to the tail of the queue. The request selected for service is that at the top of the queue, i.e. the first request to arrive is the first one served. So, there is no reordering of the queue, and it results in a random seek pattern with no optimization of head motion. However, it is easy and fair, in the sense that once a request has arrived, its place in the schedule is fixed and hence the waiting time of a request is independent of its position. The expected seek time $(T_s)_{FCFS}$ and the service time $(T)_{FCFS}$ can be expressed as [2]:

$$(T_s)_{FCFS} = S_{min} + [(D(C + 1) / 3 C^2] \quad (4)$$

$$(T)_{FCFS} = (1 / C)[(T_s)_{FCFS} + A] + [(C-1)/C][T_r /m][B] \quad (5)$$

Where $D = (S_{max} - S_{min})(C - 1)$, C = total number of cylinders ,

S_{min} = Seek time for distance of one cylinder , S_{max} = Seek time for distance of C-1 cylinders .

2-2 SSTF Algorithm

It seems reasonable to service all requests close to the current head position together , before moving the head far away to service another request .It is the basis for the shortest -seek- time- first (SSTF) scheduling technique. SSTF services requests according to their proximity to the last request serviced It tends to optimize on seek times. Thus all waiting requests are examined when service of the current request has been completed . The request that requires the shortest seek (or no seek at all) is taken from the request pool and serviced next, even if that is not the first one in the queue, and regardless of the direction in which the head must move . In this case, the innermost and outermost may receive poor service compared with the mide-range trackes. It attempts to improve the throughput by processing the request with minimum seek time, but it is at the expense of discrimination against individual requests. So, it is unfair with unacceptably high variance of the service time. The seek time $(Ts)_{SSTF}$ and the service time $(T)_{SSTF}$ can be expressed as :

$$(T_s)_{SSTF} = S_{min} + (DE - 1)/2(C - 2) \quad (6)$$

$$(T)_{SSTF} = F\,[\,(T_s)_{SSTF} + A\,] + [(1-F)(T_r/m)B] \quad (7)$$

where $E = [1/(L+1)][1+(1/(L+2))(C/(C-1))^{L+1}]$, $F = [(C-1)/C]^{L'}$ and

L' = mean number of requests served in one sweep across the surface (practically $1.5 < L'/L < 1.72$)

2-3 SCAN Algorithm

It was developed to overcome discrimination and high variance in response times of SSTF. It chooses the rwquest that results in the shortest distance in a preferred direction. The head moves, servicing all requests in that direction . It does not change direction until there are no further requests pending in that direction or until it reaches the end . Then the head is reversed and servicing continue. Thus , with *scan* the head acts as a shuttle as it sweeps back and forword access all trackes, serving the requests. It tends to optimize on seek times . The expected seek time $(T_s)_{SCAN}$ and the service time $(T)_{SCAN}$ can be formulated as:

$$(T_s)\ SCAN = S_{min} + (S_{max} - S_{min})\,/(1 + L') \quad (8)$$

$$(T)\ SCAN = F\,[(Ts)SCAN + A\,] + [\,1\text{-}F][Tr/m\,]B \quad (9)$$

3- SIMULATION MODEL

To obtain good performance from a computer system, it is necessary to investigate the performance of ll components of this system. One of the most important components is the disk scheduling algorithm the operating system. The essential qualties of a good scheduler are minimum values of the ving times : service, waiting, turnaround, minimum value of variance, and maximum value of put. The three chedulers (FCFS,SSTF,and SCAN) described above are chosen as representative the next step of investigation is to determine which scheduler satisfies all these essential qualities. This section is concerned with the performance evaluation of these schedulers, to differentiate among them over wide range of operation conditions. Thus, extensive computer simulation runs have been carried out. The simulation model is given in figure (1). It is composed of the following four parts;

(1)Disk characteristics: the disk is characterized by the number of cylinders, heads,block size, and block per track.

(2) *Request generation:* the request is randomly generated and simulated by service type, request length, and address (cylinder, track, starting record). According to the number of requests per unit

time, the random function is based on three levels of requeste; Low(from 0 to 25), moderate (from 25 to 50) and heavy (from 50 to 100).

(3) Scheduler: It is to apply a disk scheduling (FCFS or SSTF or SCAN)to serve the requests.

(4) Evaluation: It is to evaluate the following performance cerıteria; throughput, waiting time, turnaround time, mean response time, and the variance.

The simulation assumes that the delay is introduced and queues are formed only out a disk. If several devices contend for a chhannel, queues may form for the channel. Likewise,if several drives are attached to one controller, queues may form at the controller. Consideration of the effects of these additional contentions is outside the scope of this paper. Therefore, our model is based on the service time of channel and control unit are not considered. The schedulers are programmed in C language and tested using VMS (Variable Memory System) operating system, on MVAX (Micro Virtual Adress eXtension) computer. Our departement system, MVAX, is equiped with the following : main memory of 4MB, virtual memory space of 4GB, . dual diskette drive of 400KB, magnetic tape drive of 95MB, Line printer , . 18 terminal lines and 2 fixed winchestr disk drives; each of 71MB. The disk parameters are : .average seek time = 30 m.sec, average Latency time = 8.33 m.sec average access time = 38.33 m.sec, and transfer rate = 184.32 KB/sec.

4- RESULTS

To compare the performance of disk scheduling algorithms (FCFS).
SSTTF and SCAN),a series of simulation experiments are carried out The experimental results are summarized in figure (2). The goodness of the schedulers has been evaluated according to the performance indicators of that figure. Let us Look at the differences among these algorithms for these indicators. The First algorithm, FCFS, is easy to program and intrinsically Fair. so, it appears to be the most commonly used scheduler. However, it may not provide the best service, because it is acceptable only when the load on the disk is low, but as load grows it tends to saturate the device and response times become Large. It shows that SSTF results in better throughput rates than FCFS, and mean response times tend to be Lower for moderate Loads. At Low requests level, SSTF gives results indistinguished from FCFS. Thus, the SSTF, while a substantial improvement over FCFS is not optimal. One significant drawback is that higher variances occur on response times because of the discrimination against the outermost and innermost track. When the significant improvements in throughput and mean response times are considered , this increased variance may be tolerable. So that SSTF is useful in batch processing systems, and it is unacceptable in interactive systems . At Low Load, SCAN is inefficient because of the necessity to sweep across the disk regardless of the queue size . The simulation of SCAN shows that as the Load on the disk queuer increases, the difference in the performance SCAN and SSTF is nearly distinguished. Because of the oscillating of the head in SCAN . the outer tracks are visted less often than the mid-range tracks. But, this is not as serious as that of SSTF ; and SCAN eliminates much of these discrimination and offers much lower variance. One aspect of these two techniques SSTF and SCAN, is that mean seek distance decreases as Load Increases . Increase in arrival rate lead to increase in mean queue lenght , hence reduction in mean

seek distance, With a corresponding reduction in service time. In consequence, under high loads, SSTF and SCAN ore more stable than FCFS. From the point of view of waiting time , SSTF and SCAN are nearly identical. At Low input Load SSTF has a Lower variance than SCAN , while the latter is preferrable for the heavy Load .

5- CONCLUSION

We consider three various schedulers for improving disk scheduling efficiency, and compare and constrat the schedulers in the context of several performance criteria .The simulation results show that SSTF and SCAN offer significant advantages in turnaround time, waiting time ,variance, and throughput, compared to that of FCFS. While, under loading conditions, FcFs is an acceptable way to service requests, which is easy to implement. The results of SSTF and SCAN are almost nearly identical. If the throughput is of Sole concem, then we must select SSTF. But, SSTF is unacceptable in interactive systems, becuase the interactive users care more about tha variance of waiting time than they do about mean. At the high input load SCAN is efficient over SSTF. However, it is important to realize that the benefits, to be gained from the more complicated schedulers, are not as great as might be expected. Also, the effect of relatively small reduction in means seek distance is even more noticeable in newer disk drives, because manufacturers have been more successful in reducing seek times than in increasing rotation speeds. It is hoped that our new suggestion can form the basis of a new paper, its results are now almost complete and may be reported in the near future.

D

6- REFERENCES

(1) Denning, P.J., " Effects of Scheduling on File Memory Operations ", Proc. AFIPS SJCC. Vol.30 pp.9-21, 1967.

(2) Weingarten ,A .,"The Analytical Design of Real-Time Disk Systems," Proc.IFIPS Congr., pp. D131-D137 1968.

(3) Frank, H. , " Analysis and Optimization of Disk Storage Devices for Time-Sharing Systems." JACM, Vol. 16, No.4 , p.602-620, 1969

(4) Deitel, H. M. , " An Introduction to Operating Systems ", Addisen-Wesley, 1983.

(5) Daniel, S. and Geist, R., " V-Scan: An Adaptive Disk Scheduling Algorithm, " Proc, of the IEEE int. workshop on comp. sys. org., New Orleans, LA, U.S.A. pp 96-103, March 1983.

(6) Silberschatz, A. andJ. Peterson, " operation system concepts " Addison-Wesley, 1985.

(7) MicroVAX II "Technical manual", digital equipment corporation (DEC), U.S.A, 1985.

(8) Coffman,E., "Personal Communication",May 1986.

(9) Geist, R. ; Reynolds, R. and Pittard, E., "Disk Scheduling In System V" , ACM Sigmetrics Conferance on Measurement and Modeling of Computers Systems, Banff, Alta, Canda, Vol.15, No.1, pp 59-68, 1989 May

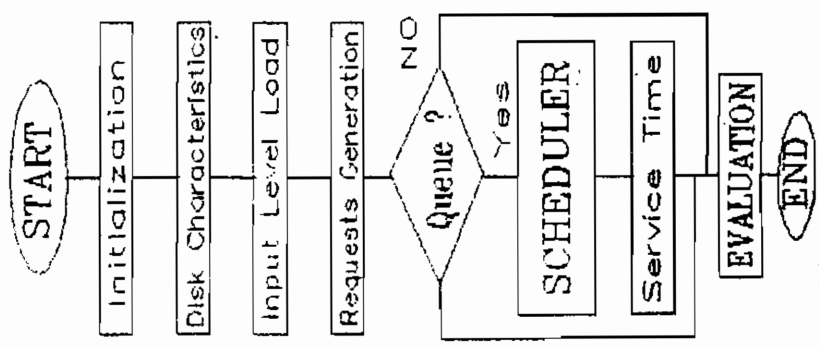
(10) Daniel, S. and Geist, R., " A Continum of Disk Scheduling Algoritms, "ACM TOCS, Vol.5, 1987.

(11) Manolopuos, Y. P. and Kollias, J. G., " Estimating Disk Head Movement In Batched Searching, " Bit (Denmark), Vol.28, No.1, pp 27-36, 1988.

(12) Wang C.C and Weems, " Some Considerations In using VSCAN Disk Scheduling With Optimal page Arrangements ", ACM Sigmetrics Conference on measurement and modeling ofComputer systems, Santa fe, NM, U.S.A., pp 27, May 1988.

Req. Per. U.T.	Performance Indicators											
	Turnaround Time			Waiting Time			Variance			Throughput		
	FCFS	SSTF	SCAN	FCFS	SSTF	SCAN	FCFS	SSTF	SCAN	FCFS	SSTF	SCAN
10	8.0	6.0	6.0	6.5	4.0	4.0	2.3	1.7	1.7	10.0	10.0	10.0
15	9.5	7.0	7.0	8.0	5.5	5.5	2.6	2.1	2.1	13.0	13.0	13.0
20	10.5	7.5	7.5	9.0	6.0	6.0	2.8	2.3	2.3	17.0	17.0	17.0
25	11.0	7.8	8.0	9.6	6.5	7.2	3.0	2.4	2.4	20.0	23.0	20.0
30	12.0	8.0	8.5	11.0	7.1	8.0	3.2	2.4	2.4	21.0	25.0	25.0
40	16.0	11.0	10.0	13.5	9.0	9.0	3.6	2.6	2.7	27.0	40.0	40.0
50	18.0	12.0	11.0	16.0	10.0	9.5	4.0	2.8	2.8	35.0	50.0	47.0
60	19.0	13.0	12.0	17.0	10.3	10.0	4.2	2.9	3.0	43.0	57.0	57.0
70	20.5	13.0	12.0	17.0	11.0	10.0	4.5	3.1	3.0	53.0	63.0	68.0
80	21.0	14.0	13.0	18.0	12.0	11.3	4.8	3.2	3.1	60.0	69.0	73.0
90	21.0	15.0	13.0	18.1	13.0	11.5	5.1	3.3	3.1	60.0	70.0	75.0
100	22.0	16.0	14.0	19.0	14.0	12.5	5.4	3.4	3.2	60.0	70.0	75.0

Figure(2) Simulation Results



Figure(1) Simulation Model