Observing the Effects of Multi-Zone Disks *

Rodney Van Meter
Information Sciences Institute
University of Southern California

Abstract

Current generations of hard disk drives use a technique known as **zoned constant angular velocity** (ZCAV), taking advantage of the geometry to increase total disk capacity by varying the number of disk sectors per track with the distance from the spindle. A side effect of this is that the transfer rate also varies with sector address. We analytically estimated and measured this effect on file system performance on a BSD Fast File System, showing a drop of roughly 25% in peak transfer rate depending on head position. We also show that, while ZCAV effects cannot be ignored, a simple linear model adequately estimates the performance from the few parameters normally available in disk drive spec sheets.

1 Introduction

Many magnetic disk drives use a technique known as *zoned constant angular velocity* (ZCAV), taking advantage of the geometry to increase total disk capacity by varying the number of disk sectors per track with the distance from the spindle. A side effect of this is that the transfer rate also varies with block address.

Despite some excellent recent work on modeling the behavior of disk drives [10, 14], the effects of ZCAV have generally not been taken into account in the design of file systems. Worthington et al [13] built a disk model which includes zone information, but the emphasis of their work is on disk scheduling algorithms to reduce latency, rather than improve throughput. Ghandeharizadeh has suggested [4] that file placement be adjusted based on access history to take advantage of ZCAV effects, but no work has measured the effects directly. The Microsoft Tiger Video Server [2] uses a simple placement algorithm in which primary data is placed on outer tracks and secondary (redundant, infrequently-accessed) data is placed on inner tracks.

We estimated and measured the effect of ZCAV on file system performance on a BSD fast file system, showing a drop of roughly 25% in peak transfer rate depending on head position. We also show that, while ZCAV effects cannot be ignored, a simple linear model adequately estimates the performance from the few parameters normally available in disk drive spec sheets.

The rest of the paper is organized as follows. ZCAV is explained in detail in section 2. In section 3 we extract the zoning information for the disk drive used in our experimental analysis. In the following section an analytic model for estimating ZCAV drive performance is presented. Then, our experimental setup is described, followed by our measured results and conclusions.

2 Zoned Constant Angular Velocity

Magnetic disk drives consist of one or more rotating platters on a common spindle. Data is written and read by magnetic *heads*, generally one per surface (often with a spare surface, so that the number of heads is one less than twice the number of platters). A *track* is a concentric circle on one surface. The collection of tracks at the same distance from the spindle on each surface constitute a *cylinder*. A track consists of a number of *sectors* (occasionally called *blocks*), the smallest unit of data that can be read or written by the drive (typically 512 or 1024 bytes, but theoretically any number). The triple `<cylinder,head,sector>` uniquely defines a location on the drive. See [10, 13] for good introductions to disk architecture.

ZCAV is a technique adopted by hard disk manufacturers to increase the capacity of disk drives. Outer tracks, which are longer, contain more sec-

*This research was sponsored by the Advanced Research Projects Agency under Contract No. DABT63-93-C-0062. Views and conclusions contained in this report are the authors' and should not be interpreted as representing the official opinion or policies, either expressed or implied, of ARPA, the U.S. Government, or any person or agency connected with them.

tors than the shorter inner tracks. The cylinders are grouped into *zones* that all have the same number of sectors per track. Some manufacturers refer to this as *Zoned Bit Recording, ZBR*. It is referred to as *notches* or a *notched drive* in the Small Computer Systems Interface (SCSI) specification [1].

As a side effect of this, since the time per rotation is constant, the number of sectors read per second (and hence the transfer rate) is higher on outer tracks. The read and write electronics must be able to keep up with the higher data rates required.

Compact disks (hence, CD-ROM) and old 400KB and 800KB Macintosh floppy drives achieved similar increases in density by varying the rotation speed to achieve *constant linear velocity*. For high-performance hard disk drives this is impractical, since each seek also means fighting high angular momentum to reach the correct speed, increasing the latency on seeks to an unacceptable level.

As table 1 shows[1], the transfer rate of the outer zones of current disk drives from a major manufacturer exceeds that of the inner zones by factors ranging from 1.45 to 1.9. It is interesting to note that the disks with the highest capacity are not necessarily those with the highest ratio of inner to outer transfer rate.

The ST31200, for example, falls off from 47.2 to 26.8 Mbps, a drop of 43%. Thus, if the disk is operating mainly in the inner regions of the disk, performance can be expected to fall to just over half of the peak rate. Although not generally stated in the user manuals for the disk drives, empirical evidence indicates that the lower-numbered blocks (for a SCSI command set interface) are stored on the outer tracks.

Note that these transfer rates are internal preformat transfer rates; we will use this information to calculate the user data rate in the next section.

Manufacturers sometimes report an "average" number of sectors per track for ZCAV disk drives. This number appears to be arrived at by totalling the number of sectors in the drive and dividing by the number of tracks. It does not attempt to reflect the fact that a higher percentage of the sectors are in tracks with more sectors. This average is useful for filling in the BSD disk format information (see the manual pages for `fs` and `newfs`), which retains the cylinder, head, sector model.

[1] Most of these values were retrieved from Seagate's web site (`http://www.seagate.com`), but the availability of data there varies.

3 Determining Zone Information

SCSI is a commonly used interface for disk drives, and all of the drives we deal with in this paper have SCSI interfaces. At the SCSI command level, sectors are referred to by a logical block address, which the device controller maps to a physical location.

Some information about the disk geometry is often available through the `MODE SENSE Notch and Partition Page` on SCSI disk drives. This page reports the number of notches. The two drives used for this paper, the ST31200 and the ST11200, both report 23 notches in this page. On some drives it is possible to read some information about each zone using `MODE SELECT` and `MODE SENSE`. However, not all drives implement this functionality. The ST31200 supports this, but the ST11200 does not. The ST31200 only reports the number of cylinders in a zone, however, not the number of sectors per track or the total number of sectors in the zone.

More detailed information can be obtained by using `SEND DIAGNOSTIC` and `RECEIVE DIAGNOSTIC` with the `TRANSLATE ADDRESS` page. This provides the cylinder, head and sector number for each logical block, allowing easy determination of the number of sectors on a track, as well as two other important performance factors: the delay incurred by switching tracks and by switching cylinders, measured in sectors. It is interesting to note that on a Sparc 20/51 each address translation takes roughly 50 milliseconds, clearly at least one order of magnitude more than the actual translation requires. The reason for this delay is currently unknown.

The intratrack instantaneous transfer rate can be determined by multiplying the number of bytes per track by the revolutions per second,

$$\frac{bytes}{track} * \frac{revs}{second}$$

To find the sustained user rate for long transfers, this must be multiplied by the factor

$$\frac{h * s}{h * s + (h-1) * g_t + g_c}$$

where h is the number of heads (tracks per cylinder), s is the sectors per track, g_t is the track-switch skew (gap) (measured in sectors) and g_c is the cylinder-switch skew (also in sectors). When reading continuously, the drive executes $h-1$ track switches plus one cylinder switch, per cylinder read.

Table 2 gives the detailed zone information for the Seagate ST11200 used in these experi-

Drive	capacity (GB)	min internal xfer rate (Mbps)	max internal xfer rate (Mbps)	ratio
Barracuda ST11950	1.69	34.3	56.5	1.65
Barracuda ST32171	2.25	75	120	1.60
Elite ST43400	2.9	35	52	1.49
Decathlon S5850A	0.71	32.45	61.65	1.90
Hawk 4 ST15230	4.29	34	61	1.79
Hawk 2XL ST31051	1.05	44	66	1.50
Elite ST410800	9.09	44	65	1.47
ST31200	1.06	26.8	47.2	1.76
ST11200	1.05	23.2	40.6	1.75

Table 1: Transfer Rates for a Variety of Seagate Disks

zone	start	cyls	heads	sec/trk	zonesec	totsec	MB/sec.	trotgap	crotgap	adjMBs
1	0	205	15	94	289050	289050	4.34	18	28	3.62
2	205	30	15	93	41850	330900	4.29	17	28	3.61
3	235	93	15	92	128340	459240	4.25	17	27	3.56
4	328	33	15	91	45045	504285	4.20	17	27	3.52
5	361	68	15	88	89760	594045	4.06	17	26	3.39
6	429	144	15	84	181440	775485	3.88	16	25	3.24
7	573	38	15	83	47310	822795	3.83	16	25	3.19
8	611	78	15	80	93600	916395	3.69	15	24	3.09
9	689	79	15	77	91245	1007640	3.56	15	23	2.96
10	768	120	15	76	136800	1144440	3.51	15	23	2.91
11	888	81	15	75	91125	1235565	3.46	14	22	2.90
12	969	41	15	74	45510	1281075	3.42	14	22	2.86
13	1010	80	15	73	87600	1368675	3.37	14	22	2.81
14	1090	79	15	71	84135	1452810	3.28	14	21	2.72
15	1169	157	15	65	153075	1605885	3.00	13	20	2.49
16	1326	178	15	62	165540	1771425	2.86	12	19	2.38
17	1504	35	15	61	32025	1803450	2.82	12	19	2.34
18	1539	168	15	57	143640	1947090	2.63	11	18	2.19
19	1707	82	15	56	68880	2015970	2.59	11	17	2.15
20	1789	80	15	54	64800	2080770	2.49	11	17	2.06

Table 2: Extracted Zone Information for ST11200 with Calculated Transfer Rates

ments. This data was obtained by a modified version of John DiMarco's `scsiinfo`, using a `SEND DIAGNOSTIC/RECEIVE DIAGNOSTIC RESULTS` command pair with the `TRANSLATE ADDRESS` page for each block on the disk, then hand-extracting the zone boundaries. On disks that also support setting the active notch on the `MODE SELECT Notch and Partition Page`, it is possible to more directly extract the cylinder boundaries. The ST31200, for example, returns the notch size in cylinders, but not the total sectors in the notch. Determining the notch boundaries is also complicated by the track skew, sparing of sectors, and sector remaps. The capacity of each zone as listed does not take into account remapped or skipped sectors.

In table 2, the first column is zone number, starting from the outer edge, in accordance with block numbering. *start* is the cylinder number for the start of the zone. *cyls* is the number of cylinders in the zone. *heads*, the number of data heads used, is constant for the whole disk drive. *sec/trk* is the number of sectors per track in the zone. *zonesec*, the total number of sectors in the zone, is the product of the prior three columns; *totsec* is a running total of the zonesec column. The columns in the table labeled *trotgap* and *crotgap* are the track and cylinder skew. *MB/sec.* is the intratrack transfer rate determined as above, and *adjMBs* is the rate adjusted by the track and cylinder skew, as above. As the table shows, the reduction in transfer rate caused by fewer sectors in a zone can sometimes be almost completely offset by a reduction in the track skew. Compared with the values of 23.2 to 40.6 Mbps internal transfer rates cited in the manufacturer's manual, the adjusted values are 29% lower, and represent reasonable "not to exceed" values for system transfer rates. It is also worth noting that the drive reports 23 notches on the notch and partition page, but only twenty were discernable from the logical to physical block map. The transfer rate is graphed in figure 1, which is explained in detail in section 4.

4 Analytic Approach to Estimating Performance

In this section, we consider three abstract examples, then analyze the disk drive used for the experiments. Transfer rates here are quoted in sectors per revolution; multiplying by revolutions per second and bytes per sector (both constants) would give bytes/second.

The first example is a hypothetical three-zoned disk drive. The outer zone is 100 tracks of 175 sectors, the middle zone is 100 tracks of 137 sectors, and the inner zone is 100 tracks of 100 sectors. This is overly simplistic but the ratios are common. The total capacity of the drive is $100*175+100*137+100*100 = 41200$ sectors. Roughly 42% of the sectors are in the outer zone, 33% in the middle zone, and 24% in the inner zone. Figure 2 shows the transfer rate in each zone versus track number. Figure 3 plots the transfer rate versus block number. Note the different position of the boundary between zones relative to figure 2, due to the higher capacity of the outer zones.

The "average" number of sectors per track is 137. If we assume that each sector is accessed with equal frequency, the "average" transfer rate is $(17500 * 175 + 13700 * 137 + 10000 * 100)/41200$, or 144 sectors/revolution, due to the higher probability of being in a high-sectors-per-track zone. This effect alone leads to an error of 5% when estimating performance based solely on the mean number of sectors per track.

As a second example, consider a more finely-grained zoning. Let the disk drive consist of one track of 175 sectors, one of 174 sectors, etc. down to an inner track of 100 sectors, for a total capacity C of

$$C = \sum_{i \in Z} s_i * t_i$$

where s_i is the number of sectors per track in zone i, t_i is the number of tracks in the zone, and Z is the set of zones. In this case, $s_i = 175 - i$ and $t_i = 1$, so this reduces to

$$C = \sum_{i=100}^{175} i = 10,450$$

sectors. The mean number of sectors per track is $10,450/76 = 137.5$

Figure 4 shows transfer rate versus track number. Figure 5 shows the transfer rate versus block address for this example. Visually, it is nearly linear. A small n^2 factor would be expected to cause the transfer rate to fall off more quickly at higher block numbers (fewer sectors per track mean fewer sectors per zone, meaning the advance to yet-smaller zones accelerates), as shown in figure 5. However, this factor appears to be unimportant, to first order.

The median transfer rate R_{med} is the transfer rate of block number $C/2$. In this case, block 5225 is on track 143, so it has a transfer rate of 143, 4% higher than the mean sectors per track.

The "average" transfer rate, again assuming equal probability of access for each sector, would be the

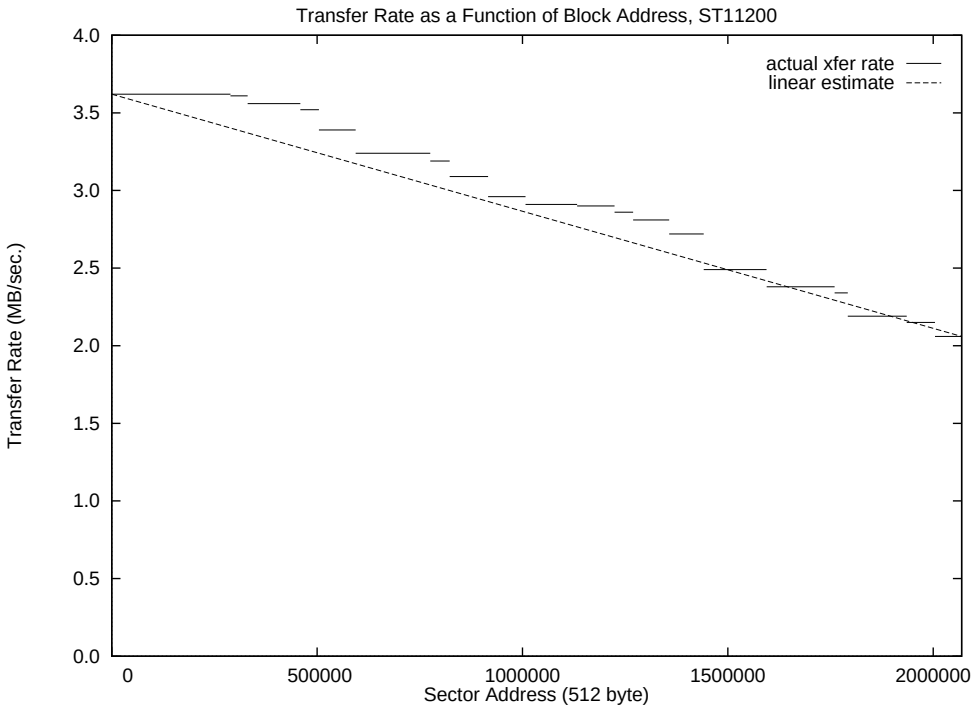

Figure 1: ST 11200 Zones with Calculated Transfer Rates

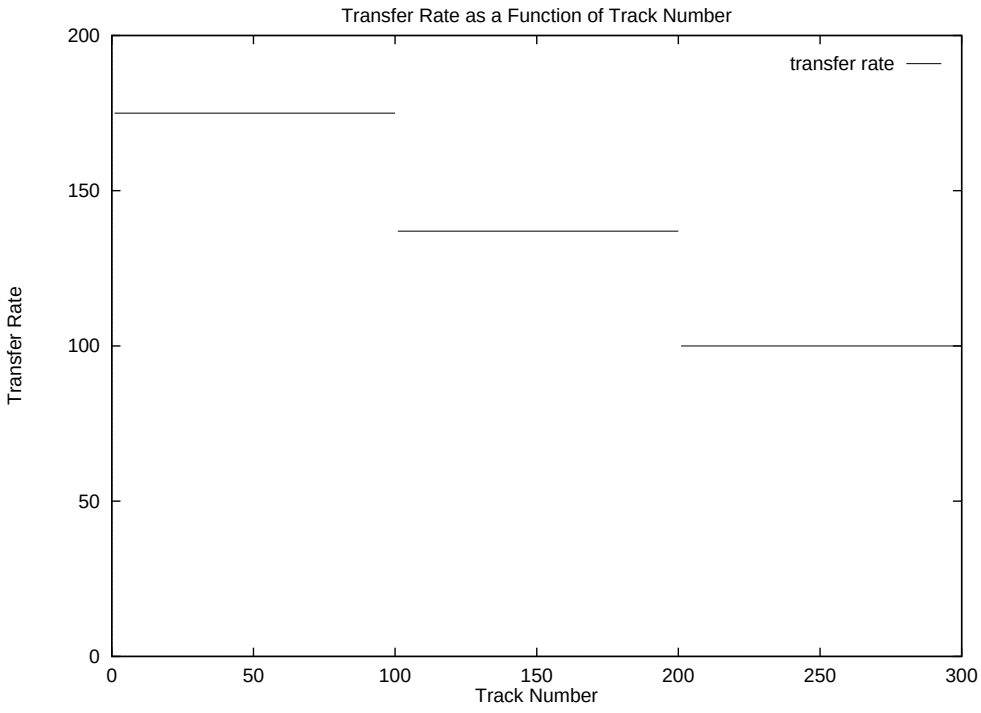

Figure 2: Three large zones, transfer rate v. track no.

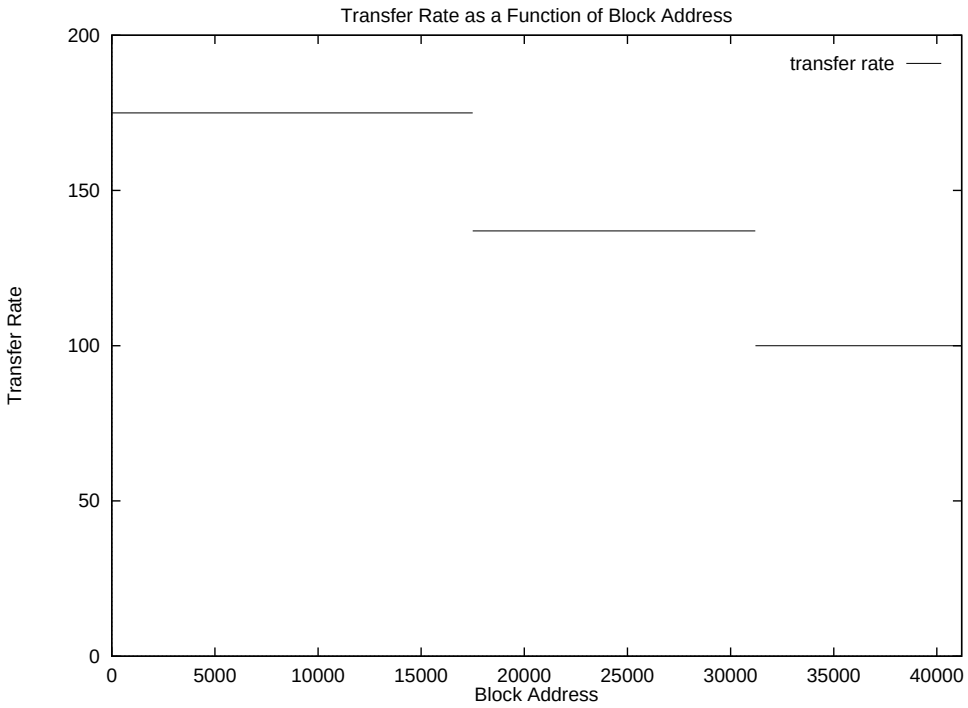

Figure 3: Three large zones, transfer rate v. block no.

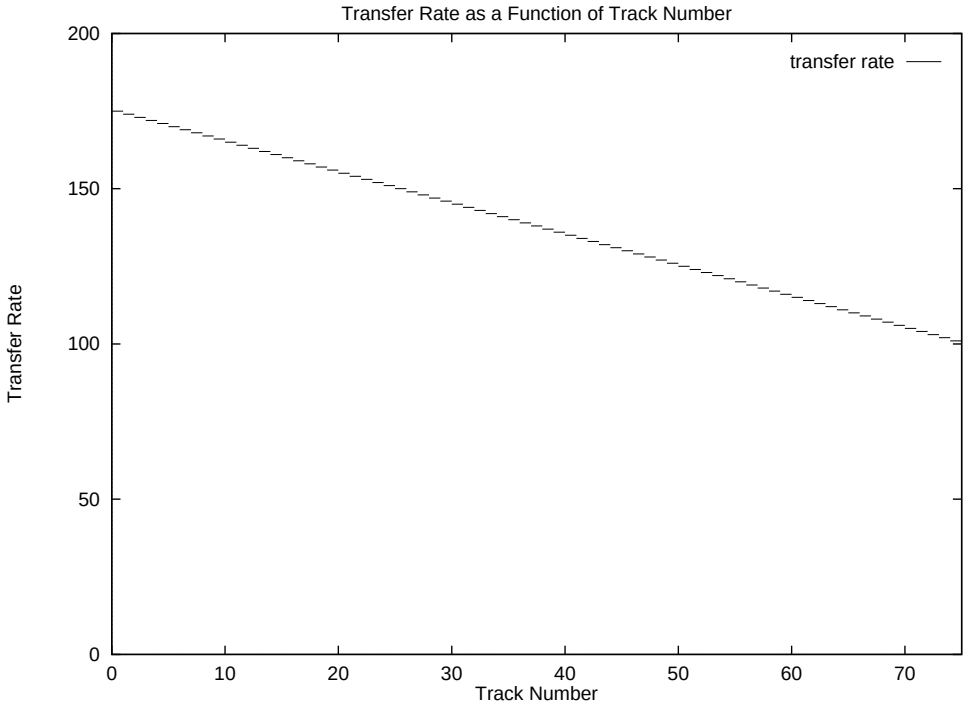

Figure 4: Single-track Zones, transfer rate v. track no.

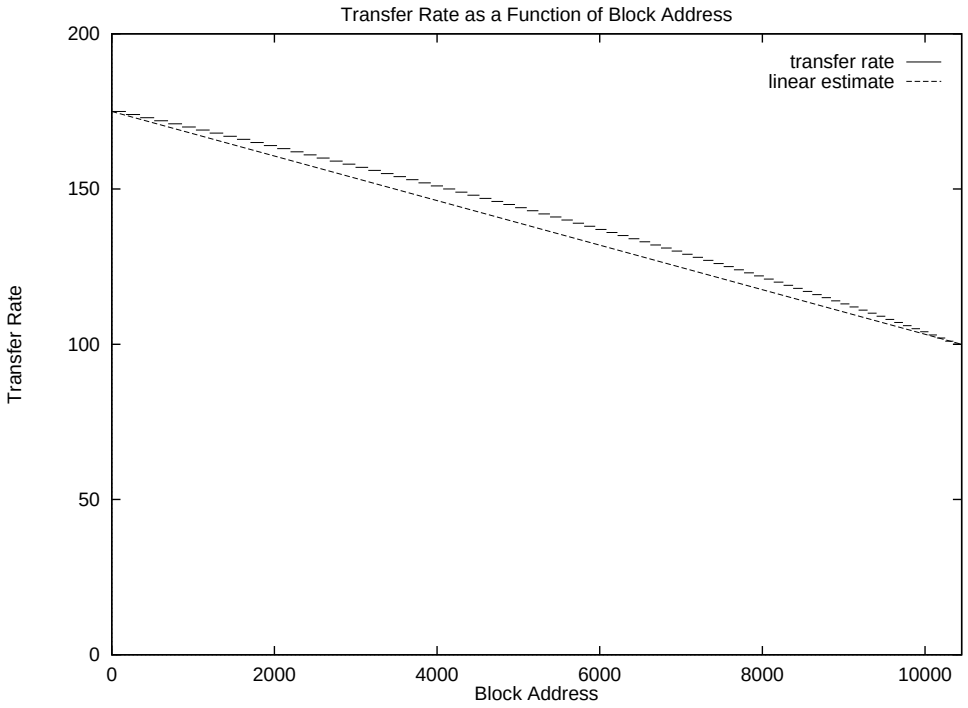

Figure 5: Single-track Zones, transfer rate v. block no.

sum of the transfer rates for each of the individual blocks, divided by the total number of blocks:

$$\begin{aligned} R_{ba} &= \frac{\sum_{i \in Z} s_i^2 * t_i}{C} \\ &= \frac{\sum_{i \in Z} s_i^2 * t_i}{\sum_{i \in Z} s_i * t_i} \end{aligned}$$

This will *not* be equal to the "average" sectors per track reported by the manufacturer, times the rotations per second:

$$\frac{avg.bytes}{second} = \frac{bytes}{sector} * \frac{avg.sectors}{rotation} * \frac{rotations}{second}$$

In our example 2, R_{ba} simplifies to $\sum_{i=100}^{175} i/C = (1,801,800 - 328,350)/C = 1,473,450/10,450 = 141$, a very modest 2.5% increase from simply assuming it to be the mean of the max and min transfer rates. For all practical purposes, therefore, we can estimate the mean transfer rate as the mean of min and max, when the zoning is fine-grained and roughly linear with track number.

Figure 6 shows transfer rate versus block address for the Seagate ST31200, calculated based on the extracted zone information. It clearly shows the effects of more of the blocks being in the outer zones. The median transfer rate is 3.6 MB/sec, 10% higher than the 3.25 arrived at by averaging the max and min rates. The average transfer rate, assuming each sector has equal probability of being accessed, is 3.47, still 6% higher than 3.25. Again, the curve varies only slightly from linear.

Figure 1 shows the transfer rate plotted against sector address for the ST11200 used for these experiments. A simple linear estimate is also plotted, running from the transfer rate at the outermost zone to the innermost zone. This shows a rough fit, with the maximum error from the true rate being approximately 8%. Thus, while far from perfect, this exceedingly simple model is significantly more accurate than assuming a fixed transfer rate, which may vary by 40%. In addition, this can be easily estimated from the data sheets typically supplied with disk drives.

A recommended first-order estimate of transfer rate, simple enough to be implemented in a guaranteed-I/O-rate file system, would therefore be

$$R(x) = 0.7 r_{max} - \frac{0.7(r_{max} - r_{min})}{C} * x \qquad (1)$$

where C is the disk capacity and r_{max} and r_{min} are the maximum and minimum internal transfer rates reported by the disk drive manufacturer. The factor 0.7 comes from our observation in section 3

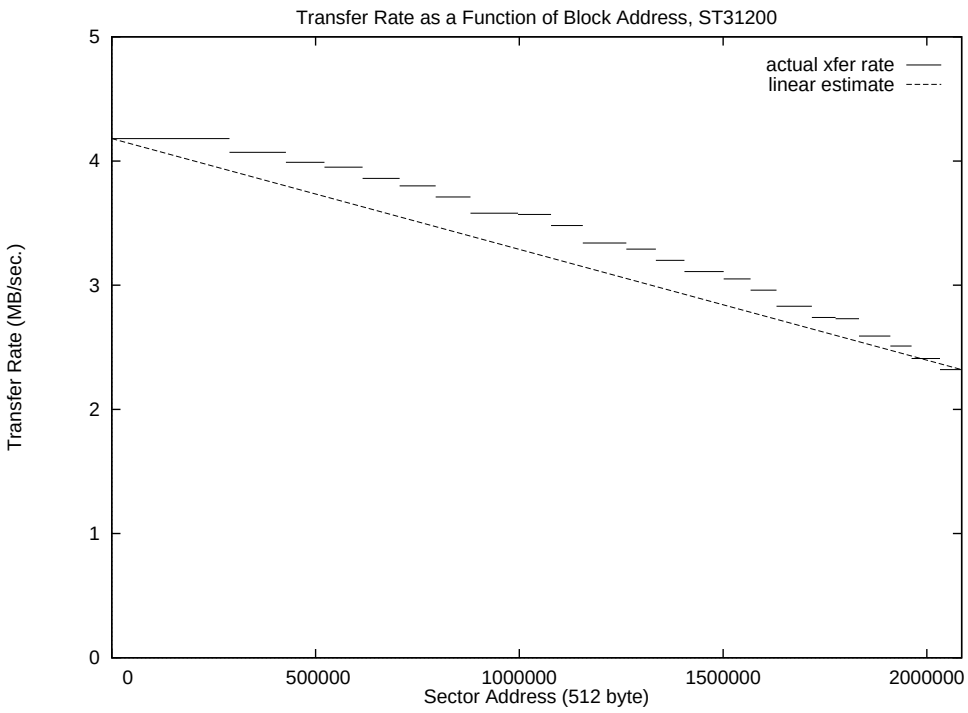

Figure 6: ST 31200 Zones with Calculated Transfer Rates

that transfer rates adjusted for sector overhead, error correction and track and cylinder skew results in a drop of approximately 29% from the manufacturer's listed transfer rates, which are instantaneous bit rates at the read/write head. Because this data is readily available, this factor can be incorporated quickly and easily by file system and device driver designers, without the necessity of tediously testing each possible disk drive. This transfer rate, of course, must be adjusted by the system's ability to sustain the I/O rate; as shown above, for a Sparc 10 running SunOS and a FFS, reads can run at device speeds, while writes run at approximately 80% of theoretical.

5 Experiment

5.1 Experimental Setup

These experiments were conducted on a Sparcstation 10 with 64 MB of main memory, and a 1.05 GB Seagate ST11200N disk drive. The actual bandwidth of this disk drive, as shown in table 2, varies from approximately 2.06 to 3.62 MB/sec., a factor of 1.75. Any read or write rate that exceeds that has clearly been the beneficiary of caching, either the file system's buffer cache or the disk drive's data block cache. According to the manual [11], this disk drive has 23 zones, or notches, and an average (mean) of 73 sectors per track, 15 heads, 1,872 cylinders for a total of 28,080 tracks. The drive rotates at 5,411 rpm. Write caching at the disk is disabled; all writes are synchronous.

The file system is a SunOS UFS, essentially a BSD fast file system [5]. The partition used for these experiments begins at sector number 655,200 and extends 687MB to the end of the disk, as reported by `dkinfo`. Thus, according to table 2, the partition starts at a transfer rate of 3.24 MB/sec. and falls to 2.06, a drop of 36%. Unfortunately, due to hardware and disk partitioning limitations, it was not possible at the time this experiment was conducted to cover the entire span of a disk.

The basic experiment runs a loop that executes a modified version of Tim Bray's `bonnie` to write a 100MB file, unmount the partition (to clear the cache and commit all modified metadata), then read the file back. Then the script records the Bonnie data file layout, deletes the file, writes a 10MB file to the system, and repeats. Thus, we have the results for 100MB written at 10MB intervals. The free space falls from approximately 610MB (user space available) to 105MB in 50 steps.

5.2 Experimental Data

When measuring the effects of the ZCAV layout on file system performance, care must be taken as numerous other factors can contribute to changes in performance. They include:

- distance from metadata (increased seek times)
- free space fragmentation
- CPU performance and system loading
- buffer cache page replacement performance

Of course, the effect on performance will vary dramatically with the file system structure, which is generally operating-system specific; this is covered in the following section.

Our data shows that the write rate varies by a factor of 1.33 (2528 KB/sec. v. 1900 KB/sec., a drop of 25%) depending on head position, even over the limited range of our experiments. Evaluating the reads is more difficult due to the high variability, but if we choose the means from the same data runs as the writes, we see a 23% drop, 3295 KB/sec. v. 2547 KB/sec. at, respectively, 608 and 270 MB free.

Figure 7 shows the mean of ten runs[2] of our 100/10 benchmark. The error bars are 90% confidence intervals. Writes are also plotted with error bars, but they are too small to see at many data points. The results clearly show a drop in performance as the disk fills, until with about 280MB free space the curve takes a sharp, unexpected upward turn.

The write values are lower than the theoretical maximum due to inevitable missed rotations. Since the disk is not allowed to cache write data, typically at least one rotation must be missed at the end of each write request. Additionally, occasionally the file system writes some metadata to the drive, requiring a seek and write, with ensuing missed rotations. The measured values are fairly consistently approximately 80% of the calculated values, indicating approximately one missed rotation in five.

Examining the layout for the data files created by the Bonnie benchmark (examined using a modified version of Keith Smith's `fsblks` utility), as shown in figure 8, confirms the hypothesis that transfer rate is related to head position, as well as providing an explanation for the upward turn near the right-hand edge. The 10 MB filler files are getting laid down with holes between them which go unused until the disk nears full.

Returning to figure 7, the points labeled *calc* are calculated from run number 9, estimating the performance by integrating the transfer rate at each block in the file, using the transfer rates calculated for each zone in section 3. It clearly shows the same features (dips and peaks) as the write and read curves. The slight difference (approximately 5%) between the read curve and the calculated estimate is because the calculated estimate does not take into account real-world overheads for command processing and latency, and CPU time in the kernel and user process. This difference (for both read and write) will be system dependent and will have to be determined empirically.

The points labeled *lin-est* are calculated using the linear estimate shown in equation 1. The largest difference from the more correctly calculated values is 7%. This error is significant, but the simplicity of this linear estimate (both in ease of determination and ease of use) may make it an acceptable substitute for detailed zone calculations. The apparent better agreement of the linear estimate than the more realistic calculation above is coincidence; the linear estimate slightly underestimates performance compared to the non-linear effects of block address and geometry as described in section 4. Note that near the disk spindle (where the curve in figure 7 dips at 280,000 KB free), the agreement between the block calculation and linear estimate is better, as we would expect.

Reviewing our concerns expressed at the top of this section, our data has good repeatability, especially on writes. The buffer cache issue has been addressed by clearing the cache via remounting the partition. Free space fragmentation proved to not be a problem. The CPU and other system components appear to be up to the task of fully utilizing the disk, clearly showing the ZCAV effects we expected.

6 Conclusions

6.1 Dependence on File System Structure

One of the interesting aspects of this work is how repeatable the data proved to be, especially for writes. This clearly demonstrated that the SunOS file allocation code depends on the current state only; the recent history of file creations and deletions does not alter future file system allocation decisions. Note that this does not mean that disk fragmentation is

[2] The runs actually used for these calculations are numbers 6 through 15; the first five represented progressive refinements of the measurement code and are discarded.

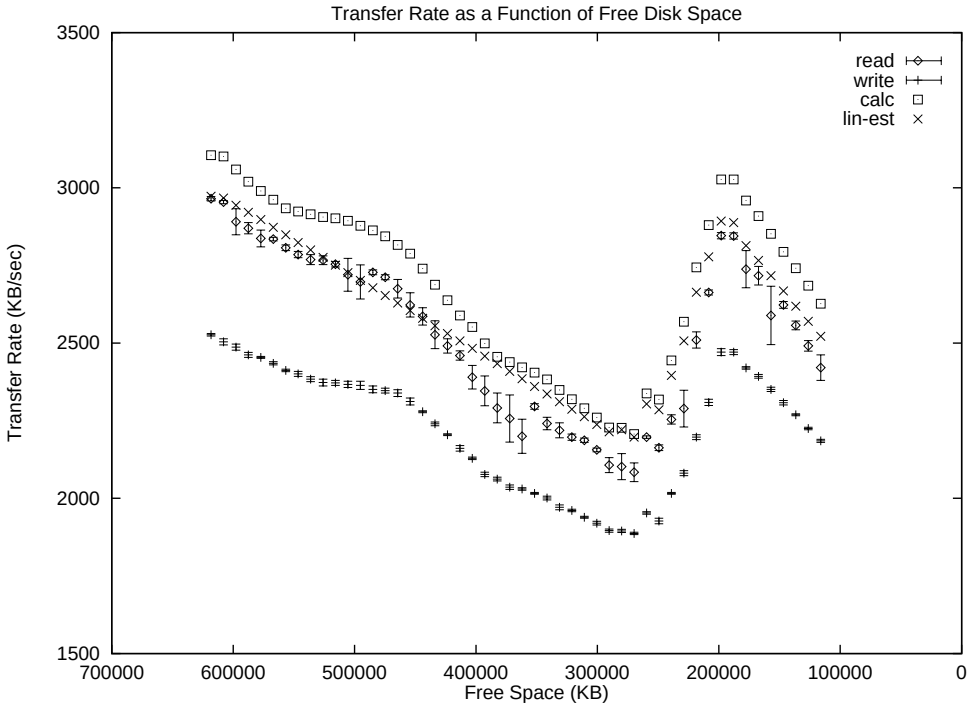

Figure 7: 10 runs of 100/10 ZCAV benchmark

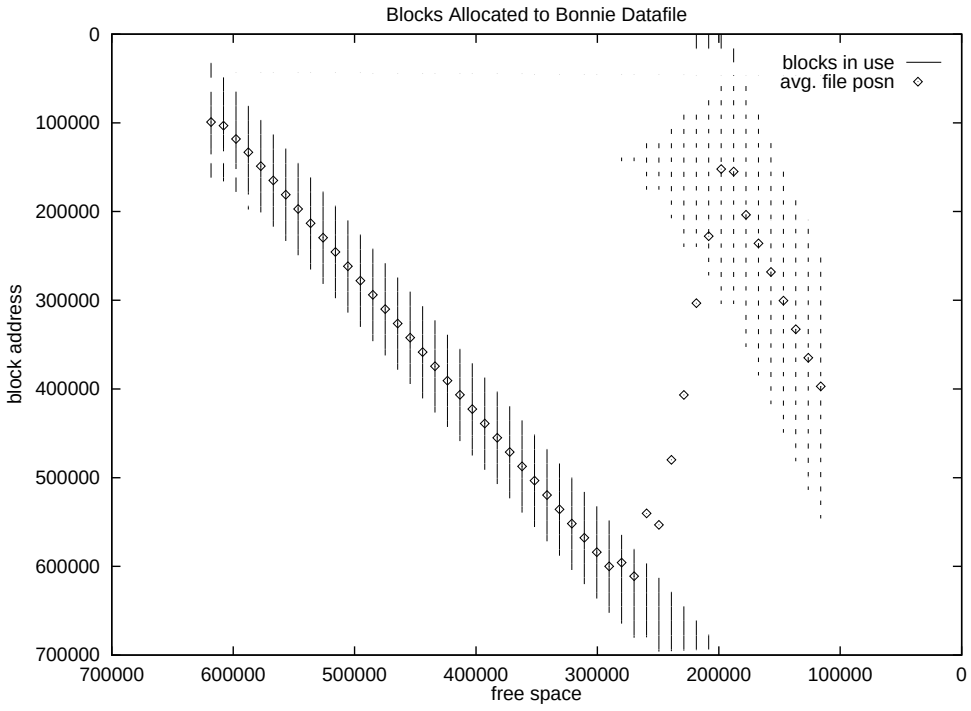

Figure 8: File layout for run # 7

not a general problem, only that once large areas of disk have been cleared of files, the reuse of that area is optimal (or at least predictable).

McVoy showed that a UFS can achieve good write performance [6][3]. The file blocks are allocated contiguously and I/Os are performed in clusters, much like an extent-based file system. Our work benefits from this work.

Other possible file system structures, such as SGI's XFS [12], may depend on more dynamic, and hence complex, data structures, and may therefore not allocate blocks as predictably. A log-based file system [9] or disk device [3] clearly will not, in their present forms, allocate blocks in a fashion amenable to improving throughput by careful choice of blocks.

6.2 Impact on File System Allocation Policies

As proposed by Ghandeharizadeh [4], the idea of including a measure of ZCAV effects into a dynamic file relocater is appealing. Such functionality could be included in a file system defragmenter, moving older, less-frequently-accessed files to lower-transfer-rate areas of the disk.

It is clear that this effect needs to be taken into account for multimedia file systems and file systems (such as SGI's XFS [12] or Rangan's multimedia ropes [8]) that provide guaranteed throughput. However, to date these have all assumed disk bandwidth is fixed, rather than a function of block address.

Larger files accessed in large chunks, for which transfer rate is likely to be more important, should be allocated to blocks at the outer edges (for a Seagate SCSI drive, the lower-numbered blocks). Small files obviously do not need to be placed in a high-transfer rate location, as their transfer time will be dominated by latency. Large files accessed in small I/O requests also will not take good advantage of the transfer rate. Determining which files will take advantage of this may require cooperation from applications, perhaps via some form of hints [7].

Incorporating knowledge of the drive's ZCAV nature into the cleaner for a log-structured file system may be useful. Data should be packed toward the spindle, so that the open area for upcoming log writes will get to use the outer, faster regions of the disk. This could be expected to improve the write performance of the LFS by 25% or more, at the expense of slower reads, in keeping with the LFS philosophy.

[3] McVoy noted, in fact, that exposing the drive's variable geometry to the system will complicate block allocation, especially in an extent-base FS.

6.3 Future Work

Obviously, we would like to try these experiments on a wider range of hardware and software platforms, especially different file systems. However, the point that performance varies with head position appears to have been adequately demonstrated. In particular, our work should be repeated with different families of disk drives from different manufacturers, to confirm both the hypothesis that ZCAV effects are user-visible, and the effectiveness of our proposed linear estimate.

Ideally, a publicly-available bank of information on drive types and zone information should be created. As more drive developers adopt standard methods of determining the zone information, of course, determining this information at boot time or file system configuration time becomes more feasible.

6.4 Conclusions

We have explained the underlying motivations behind ZCAV disk drives, and demonstrated that it does have an effect on file system performance for a BSD FFS. We have shown that it is possible, though somewhat tedious, to extract this information from at least some disk drives directly. We have proposed that a simple linear model relating transfer rate to block address should be adequate for most purposes. The measurable effect, at 23-25%, is less than the physical difference of 36% between the inner and outer disk edges of the tested partition, but still too large to ignore in performance-critical applications.

Acknowledgments

This work would have been substantially more tedious if not for the generosity of Keith Smith (`fsblks`), John DiMarco (`scsiinfo`) and Tim Bray (`bonnie`) in making their code publicly available. John Heidemann, Ted Faber and Greg Finn provided useful technical and editorial suggestions. The anonymous referees and my shepherd, Bill Bolosky, improved the paper through their comments as well.

References

[1] ANSI. *Information Technology - Small Computer Systems Interface - 2 (X3T9.2 rev 10l)*, Sept. 1993.

[2] W. J. Bolosky et al. The tiger video fileserver. In *Proc. Sixth International Workshop on Network and Operating System Support for Digital Audio and Video*, Apr. 1996. available at ftp://ftp.research.microsoft.com/pub/tech-reports/Winter95-96/TR-96-09.ps.

[3] W. de Jonge, M. F. Kaashoek, and W. C. Hsieh. The logical disk: A new approach to improving file systems. In *Proc. Fourteenth ACM Symposium on Operating Systems Principles*, pages 15–28, Dec. 1993.

[4] S. Ghandeharizadeh, D. J. Ierardi, D. Kim, and R. Zimmerman. Placement of data in multi-zone disk drives. rcvd from author, June 1995.

[5] S. J. Leffler, M. K. McKusick, M. J. Karels, and J. S. Quarterman. *The Design and Implementation of the 4.3BSD UNIX Operating System*. Addison-Wesley, 1989.

[6] L. W. McVoy and S. R. Kleiman. Extent-like performance from a UNIX file system. In *Proc. 1991 USENIX Winter Technical Conference*, pages 33–43. USENIX, 1991.

[7] R. H. Patterson, G. A. Gibson, E. Ginting, D. Stodolsky, and J. Zelenka. Informed prefetching and caching. In *Proc. 15th Annual ACM Symposium on Operating Systems Principles*, pages 79–95. ACM, Dec. 1995.

[8] P. V. Rangan and H. M. Vin. Designing file systems for digital video and audio. In *Proc. Thirteenth ACM Symposium on Operating Systems Principles*, pages 81–94, Oct. 1991.

[9] M. Rosenblum and J. K. Ousterhout. The design and implementation of a log-structured file system. In *Proceedings of the 13th Symposium on Operating Systems Principles*, pages 1–15. ACM, Oct. 1991.

[10] C. Ruemmler and J. Wilkes. An introduction to disk drive modeling. *Computer*, 27(3):17–28, Mar. 1994.

[11] Seagate. *Product Manual - Hawk 1 Family SCSI-2 (Volume 1), Rev. D*, 1994.

[12] A. Sweeney, D. Doucette, W. Hu, C. Anderson, M. Nishimoto, and G. Peck. Scalability in the XFS file system. In *Proc. 1996 USENIX Technical Conference*, pages 1–14. USENIX, Jan. 1996.

[13] B. L. Worthington, G. R. Ganger, and Y. N. Patt. Scheduling for modern disk drives and non-random workloads. Technical Report CSE-TR-194-94, University of Michigan, Mar. 1994.

[14] B. L. Worthington, G. R. Ganger, Y. N. Patt, and J. Wilkes. On-line extraction of SCSI disk drive parameters. In *Proc. ACM Sigmetrics Conference*, May 1995.

Availability

The code used for these measurements and data obtained is available on the web at http://www.isi.edu/netstation/zcav/ or on the author's home page at http://www.isi.edu/~rdv/ or http://alumni.caltech.edu/~rdv/.

The author may be contacted via email at `rdv@isi.edu` or `rdv@alumni.caltech.edu`.