All-Adjacent Matrices and Their Application to Disk Shadowing

Francis C.M. Lau*
Department of Computer Science
University of Hong Kong

November 1996

Abstract

We present the combinatorial design for a kind of matrices called *all-adjacent matrices*. Such a matrix is constructed from a given set of elements; each element of the matrix is adjacent, within some column, to every other element of the set. We present the algorithm for constructing an all-adjacent matrix. These matrices have an important application—to the laying out of logical disks in a shadowed disk system so that the seek distance of read requests is minimized.

1 Introduction

Given a set of elements, $A = \{a_i\}$. The problem is to design a two-dimensional matrix, B, of dimension $m \times n$, such that every element in A is adjacent to every other element in A in one or more columns. We say that the elements satisfy the *all-adjacency requirement* and the matrix is an *all-adjacent matrix*. An element is adjacent to another element if they are in the same column and their positions within the column differ by one. Here is an example:

$$A = \{1, 2, 3, 4\}$$

*Correspondence: F.C.M. Lau, Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong / Email: fcmlau@cs.hku.hk / Fax: (+852) 2559 8447

$$B = \begin{bmatrix} 2 & 3 \\ 3 & 1 \\ 4 & 2 \\ 1 & 4 \end{bmatrix}$$

B is an all-adjacent matrix for the set A. A simple inspection can verify that: 1 is adjacent to 4 in column 1 and to $3, 2$ in column 2, *etc.* If every element in an all-adjacent matrix is adjacent to every other element in exactly one column, the matrix is a *perfect all-adjacent matrix*. B in the above example is one.

Let c be the cardinality of A. There are $\binom{c}{2} = \frac{c(c-1)}{2}$ distinct pairs of a_i's. There are $m - 1$ pairs of adjacent slots in every column of B. Therefore, for perfect all-adjacency, $n(m-1) = \frac{c(c-1)}{2}$. For the application we present in this paper, $m = c$, and hence $n = \frac{c}{2}$; that is, B is non-square and has dimension $c \times \frac{c}{2}$.

2 The Application

This work is motivated by the following practical problem in computer science.

> **Replicated disk system.** Given c logical cylinders and d identical disks (each having c physical cylinders). Map the logical cylinders to the disks so that the performance of certain disk operations is improved.

The case in which the disks maintain identical copies of the same logical cylinder sequence is called *disk shadowing* [1]. Disk shadowing with $d = 2$ is also called *disk mirroring* or *duplexing*. With data replicated in multiple disks, a read operation can be performed more efficiently. This is so because at some random point in time, the disk heads of the disks are expected to be at random positions, and the next read can be served by the disk whose head is closest to the requested cylinder. For the write operation, however, all the cylinders (one in each disk) concerned must be written, which would take a longer time than the write operation in a single-disk system on average. Bitton and Gray first analyzed the behavior of shadowed disk systems [1, 2]. Their results were later on revised by Lo and Matloff who observed that every time a write operation is performed, it destroys the randomness of the positions of the heads, and the several reads that follow will not be able to enjoy a much reduced seek time [5].

We propose in this paper a scheme in which the read operation can always be finished in optimal time. In fact, with the ideal configuration (one using $\frac{c}{2}$ disks), the head needs to travel across at most one or two cylinders. Our scheme applies different mappings from logical cylinders to physical cylinders to different disks. Our scheme should still be called a shadowed disk scheme, because the data contents in any one disk are exactly the same as those in another disk, even though the two sets of contents are organized differently. Figure 1 shows the case of $c = 8$ and $d = 4$.

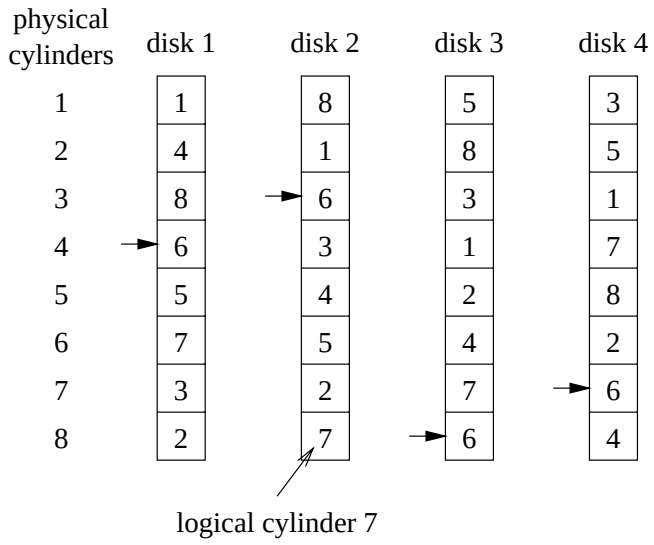


Figure 1: An 8-cylinder, 4-disk system

The important property of the scheme is that every logical cylinder is adjacent to every other logical cylinder in one of the disks. Therefore, every time after a write, we can always find a disk in which the cylinder requested by a read request that follows is either under the head or adjacent to the head. For instance, in Figure 1, after writing logical cylinder 6, a read request for logical cylinder 3 can be served by disk 2 in optimal time (crossing one cylinder). If the disks are designed to have their heads always return to the previous position after a read operation, then the system of disks will always be in the perfect situation where every logical cylinder is either directly under some head or one cylinder away from some head. This automatic returning is bounded by the seek distance of one cylinder, which should not be too great a cost to bear. In fact, automatic returning is similar in certain respects

to the anticipatory scheme proposed by King [4]. Even if automatic returning is not done, the worst performance of a read is only that of crossing two cylinders, assuming that a head is not allowed to move beyond the two neighbors (or one neighbor for cylinders at the margins) of the last logical cylinder written. We refer to the automatic returning strategy as *anchoring* and the latter strategy as *confinement*. We argue that both strategies are easy to implement in a disk's firmware.

This scheme optimizes the performance of the read operation, which however is at the expense of the write operation. The write operation becomes worse than that in normal shadowed disk systems, which is due to the fact that for any logical cylinder, one of its copies would be situated at one end of some disk; and so there is the likelihood that the two logical cylinders of two consecutive writes are $O(c)$ apart in seek distance in some disk. Still, we contend that the scheme is most useful for systems which have a large read / write ratio. Examples of such systems abound, which include many digital libraries that are accessible via the Internet.

This paper presents the algorithm for generating a perfect all-adjacent matrix of dimension $c \times d$, of which each column corresponds to the layout of logical cylinders in one of the disks in the disk array.

3 The Algorithm

Given c cylinders that are numbered from 1 to c. For perfect all-adjacency, we need $d = \frac{c}{2}$ disks. Assume d is even, and $c \geq 4$. The following algorithm generates the matrix B. A typical element of B is denoted by $B_{j,k}$, where $1 \leq j \leq c$ is the row index, $1 \leq k \leq d$ the column index. Logical cylinder i would be placed in physical cylinder j of disk k if $B_{j,k} = i$.

1. Form the $d \times d$ matrix S:

$$S_{i,j} = X_{i+j-1 \bmod d}$$

where

$$X = \begin{cases} \underbrace{2, 4, \ldots, d}_{even}, \underbrace{d-1, d-3, \ldots, 1}_{odd} & \text{if } d \text{ is even} \\ \underbrace{2, 4, \ldots, d}_{even}, \underbrace{d+1, d-1, \ldots, 1}_{odd} & \text{if } d \text{ is odd} \end{cases}$$

2. Let $k = \lfloor \frac{d}{2} \rfloor$. Form the $d \times d$ matrix F:

   Case 1—d odd, k odd:

$$F_{i,j} = \begin{cases} 1 & \text{if } S_{i,j} \text{ is odd and} \leq k \\ 0 & \text{otherwise} \end{cases}$$

   Case 2—d odd, k even:

$$F_{i,j} = \begin{cases} 1 & \text{if } S_{i,j} \text{ is even and} \leq k \\ 0 & \text{otherwise} \end{cases}$$

   Case 3—d even:

$$F_{i,j} = \begin{cases} 1 & \text{if } S_{i,j} \text{ is odd and} \leq k, \text{ and } i \leq k \\ 1 & \text{if } S_{i,j} \text{ is even and} \leq k, \text{ and } i > k \\ 0 & \text{otherwise} \end{cases}$$

3. Form the $c \times d$ matrix B:

$$\{B_{i,j}, B_{\bar{i},j}\} = \begin{cases} \{2l-1, 2l\} & \exists\, l \in \{1, \ldots, d\} \mid S_{j,l} = i \text{ and } F_{j,l} = 0 \\ \{2l, 2l-1\} & \exists\, l \in \{1, \ldots, d\} \mid S_{j,l} = i \text{ and } F_{j,l} = 1 \end{cases}$$

   where $i = 1, 2, \ldots, \frac{d}{2}$ and $\bar{i} = d + 1 - i$.

4 Some Examples

We apply the algorithm to the case of $c = 12$; we need $d = 6$ disks. The matrices S, F, and B are of dimension 6×6, 6×6, and 12×6, respectively.

1. d is even, and so the sequence X is

$$2, 4, 6, 5, 3, 1$$

   This sequence as it is equals exactly to the first column of the matrix S. The second column is a rotation (downward by one position) of the first column, and similarly for the other columns. Hence, S is as follows.

$$S = \begin{bmatrix} 2 & 4 & 6 & 5 & 3 & 1 \\ 4 & 6 & 5 & 3 & 1 & 2 \\ 6 & 5 & 3 & 1 & 2 & 4 \\ 5 & 3 & 1 & 2 & 4 & 6 \\ 3 & 1 & 2 & 4 & 6 & 5 \\ 1 & 2 & 4 & 6 & 5 & 3 \end{bmatrix}$$

2. d is even; $k = \lfloor \frac{d}{2} \rfloor = 3$, is odd. The matrix F, being superimposed on the matrix S, is as follows.

$$F \cdot S = \begin{bmatrix} 2 & 4 & 6 & 5 & \boxed{3} & \boxed{1} \\ 4 & 6 & 5 & \boxed{3} & \boxed{1} & 2 \\ 6 & 5 & \boxed{3} & \boxed{1} & 2 & 4 \\ 5 & 3 & 1 & \boxed{2} & 4 & 6 \\ 3 & 1 & \boxed{2} & 4 & 6 & 5 \\ 1 & \boxed{2} & 4 & 6 & 5 & 3 \end{bmatrix}$$

where a boxed entry corresponds to a 1 in the F matrix, and 0 otherwise.

3. Now with the matrices S and F, we can generate the final matrix, B.

$$B = \begin{bmatrix} 12 & 10 & 8 & 5 & 3 & 1 \\ 1 & 11 & 9 & 8 & 6 & 4 \\ 10 & 8 & 6 & 3 & 1 & 11 \\ 3 & 1 & 11 & 9 & 7 & 5 \\ 7 & 5 & 3 & 1 & 11 & 9 \\ 5 & 3 & 1 & 11 & 9 & 7 \\ 6 & 4 & 2 & 12 & 10 & 8 \\ 8 & 6 & 4 & 2 & 12 & 10 \\ 4 & 2 & 12 & 10 & 8 & 6 \\ 9 & 7 & 5 & 4 & 2 & 12 \\ 2 & 12 & 10 & 7 & 5 & 3 \\ 11 & 9 & 7 & 6 & 4 & 2 \end{bmatrix}$$

The first column of the matrix B can then be used for arranging the logical cylinders in the first disk, the second column for the second disk, and so on. It is easy to verify by inspection that the above matrix B is a perfect all-adjacency matrix.

Here is one more example, for the case of an odd d. Let $c = 14$; $d = \frac{14}{2} = 7$; $k = \lfloor \frac{7}{2} \rfloor = 3$. The matrix S and F combined, and the matrix B are as follows.

$$F \cdot S = \begin{bmatrix} 2 & 4 & 6 & 7 & 5 & \boxed{3} & \boxed{1} \\ 4 & 6 & 7 & 5 & \boxed{3} & \boxed{1} & 2 \\ 6 & 7 & 5 & \boxed{3} & \boxed{1} & 2 & 4 \\ 7 & 5 & \boxed{3} & \boxed{1} & 2 & 4 & 6 \\ 5 & \boxed{3} & \boxed{1} & 2 & 4 & 6 & 7 \\ \boxed{3} & \boxed{1} & 2 & 4 & 6 & 7 & 5 \\ \boxed{1} & 2 & 4 & 6 & 7 & 5 & \boxed{3} \end{bmatrix}$$

$$B = \begin{bmatrix} 14 & 12 & 10 & 8 & 6 & 4 & 2 \\ 1 & 13 & 11 & 9 & 7 & 5 & 3 \\ 12 & 10 & 8 & 6 & 4 & 2 & 14 \\ 3 & 1 & 13 & 11 & 9 & 7 & 5 \\ 9 & 7 & 5 & 3 & 1 & 13 & 11 \\ 5 & 3 & 1 & 13 & 11 & 9 & 7 \\ 7 & 5 & 3 & 1 & 13 & 11 & 9 \\ 8 & 6 & 4 & 2 & 14 & 12 & 10 \\ 6 & 4 & 2 & 14 & 12 & 10 & 8 \\ 10 & 8 & 6 & 4 & 2 & 14 & 12 \\ 4 & 2 & 14 & 12 & 10 & 8 & 6 \\ 11 & 9 & 7 & 5 & 3 & 1 & 13 \\ 2 & 14 & 12 & 10 & 8 & 6 & 4 \\ 13 & 11 & 9 & 7 & 5 & 3 & 1 \end{bmatrix}$$

5 Proof of Correctness

The strategy of the algorithm is to divide the logical cylinders (that is, the elements, $\{a_i\} = \{1, \ldots, c\}$, of the set A) into pairs. There are c elements, which are paired up into $\{1, 2\}, \{3, 4\}, \ldots, \{c-1, c\}$. The outcome of the algorithm is that any one pair will be placed adjacent to any other pair in two of the columns in B. In one of the these two cases, the order of the elements of one of the pairs will be reversed. The reason is clear when looking at the example in Figure 2. The pairs are $\{3, 4\}$

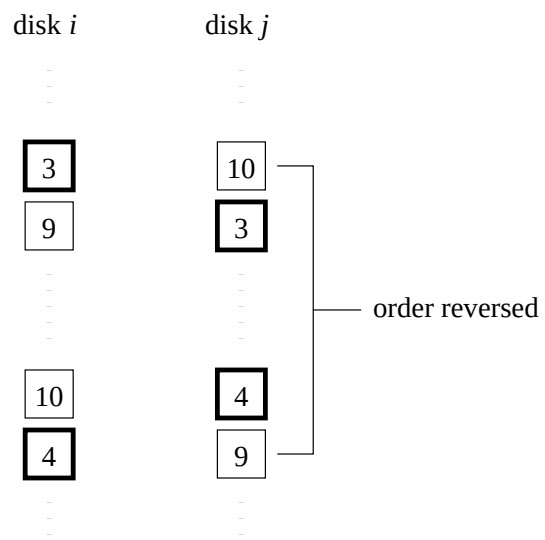


Figure 2: All-adjacency of two pairs of elements

and $\{9, 10\}$. By reversing the order of the latter in the disk j (column j in B), the four elements satisfy the all-adjacency requirement. Note that when two pairs are placed adjacent to each other, they are as shown in the figure, as opposed to having a pair being entirely on one side of the other pair. We say that these two *pairs* are adjacent and they satisfy the all-adjacency requirement. The following is obvious.

Lemma 1 *Given two pairs of elements in A, if they are placed adjacent to each other in two columns in B, and the order of one or three of the four occurrences of these elements is reversed, then the pairs satisfy the all-adjacency requirement.*

The matrix S contains the positions for each of the $\frac{c}{2}$ pairs. The first column for the pair $\{1, 2\}$, the second column for the pair $\{3, 4\}$, and so on. The matrix F then indicates which pairs (and at which positions) should have their order reversed.

We first prove that the matrix S, which is generated by Step 1 of the algorithm, does in fact make every pair adjacent to every other pair in exactly two columns in B. A number m is consecutive to another number n if $|m - n| = 1$.

Lemma 2 *Any two columns in the matrix S have exactly two row positions at which the element in one column is consecutive to the element in the other column.*

Proof: The sequence X is mapped to the first column of the matrix S; X rotated to the left by one position is mapped to the second column of S, and so on. $d-1$ rotations are needed to fill the matrix S. X has the structure as shown in Figure 3, for even d and odd d, respectively. The dashed lines connect the elements that are con-

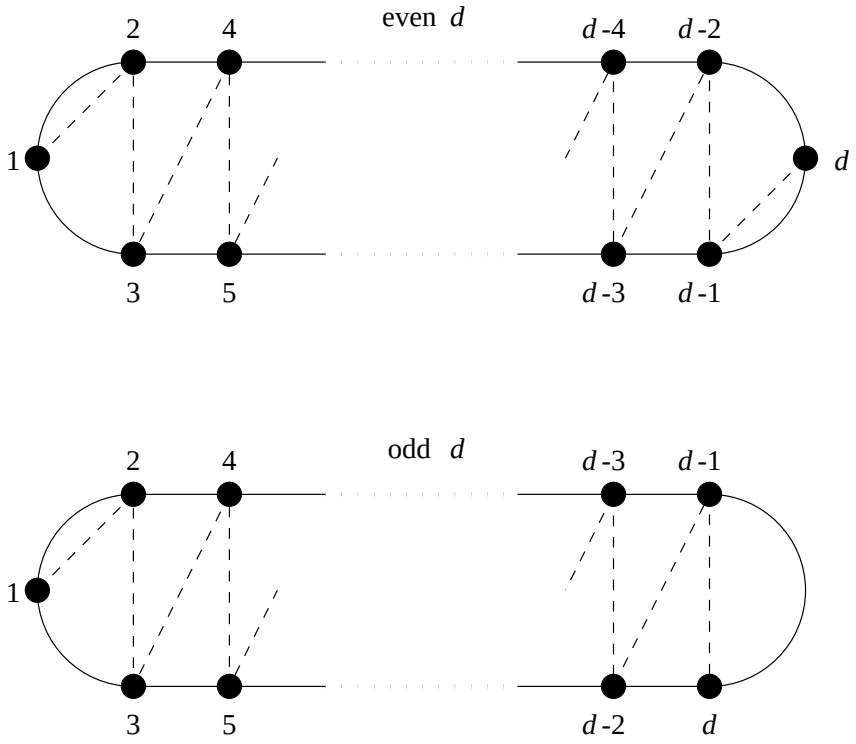


Figure 3: The X sequence

secutive. Consider the even-d case. By rotating the sequence one position to the left (*i.e.*, anti-clockwise), we move 2 to 1's previous position, and $d-1$ to d's previous position. We refer to the the rotated sequence as X^1 where the superscript denotes the number of rotations. Since X and X^1 correspond to the first and second column of the matrix S, we have two pairs of elements, $\{2,1\}$ and $\{d-1,d\}$, that are consecutive across the two columns. It can be easily seen that because of the structure of X as shown above, at every rotation, there are two such pairs that are formed;

these pairs are as follows.

$$\begin{array}{lll}
X^1: & 2 \to 1 & d-1 \to d \\
X^2: & 2 \to 3 & d-1 \to d-2 \\
\cdots & \cdots & \cdots \\
X^{\frac{d}{2}}: & \frac{d}{2}+1 \to \frac{d}{2} & \frac{d}{2} \to \frac{d}{2}+1 \\
\cdots & \cdots & \cdots \\
X^{d-1}: & d \to d-1 & 1 \to 2
\end{array}$$

where $i \to j$ means that i in the current rotation has been moved to the position of j in X, and i and j are consecutive elements.

Similarly, for the odd d case, we have the following.

$$\begin{array}{lll}
X^1: & 2 \to 1 & d \to d-1 \\
X^2: & 2 \to 3 & d-2 \to d-1 \\
\cdots & \cdots & \cdots \\
X^k: & k \to k+1 & k+1 \to k+2 \\
\cdots & \cdots & \cdots \\
X^{d-1}: & d-1 \to d & 1 \to 2
\end{array}$$

where $k = \lfloor \frac{d}{2} \rfloor$. Therefore, for either even d or odd d, there are two pairs of consecutive elements across any two columns in the matrix S. □

The two pairs of consecutive numbers across any two columns in S represent the positions in the two corresponding disks where the two pairs of logical cylinders indexed by the two columns are to be placed. Note that a column in S corresponds to a pair of logical cylinders, whereas a column in B corresponds to a disk. By Lemma 1, either one or three of these four numbers must be marked to indicate a reversed order in the placement. We next prove that Step 2 of the algorithm does exactly this; it marks one of the four numbers in the two consecutive pairs across the two columns in S.

Lemma 3 *By matrix F, the two pairs of consecutive numbers across any two columns in S have exactly one of the four elements marked.*

Proof: Recall $k = \lfloor \frac{d}{2} \rfloor$.

Case 1—d odd, k odd The two pairs of consecutive numbers across any two columns in S are one of the following.

$$\{\{1,2\},\{d-1,d\}\}$$
$$\{\{2,3\},\{d-2,d-1\}\}$$
$$\ldots$$
$$\{\{k,k+1\},\{k+1,k+2\}\}$$

Step 2 of the algorithm marks all the odd numbers that are smaller than or equal k—that is, $1,3,\ldots,k$. Clearly, for any of the above pairs, one of the four numbers involved is marked.

Case 2—d odd, k even Similar to Case 1.

Case 3—d even The pairs across any two columns in S are

$$\{\{1,2\},\{d-1,d\}\}$$
$$\{\{2,3\},\{d-2,d-1\}\}$$
$$\{\{3,4\},\{5,6\}\}$$
$$\ldots$$
$$\{\{k-1,k\},\{k+1,k+2\}\}$$
$$\{\{k,k+1\},\{k+1,k\}\}$$

Step 2 of the algorithm marks the 1's, 3's, ..., k's (if k is odd) or $(k-1)$'s (if k is even) that are in the upper half of the matrix S; and 2's, 4's, ..., k's (if k is even) or $(k-1)$'s (if k is odd) that are in the lower half of the matrix S. As a result, for any of the above pairs, one of the four numbers involved is marked. Take the pairs $\{p_1=\{1,2\},p_2=\{d-1,d\}\}$ for instance, if p_1 is in the upper half of S, its 1 will be marked; if p_1 is in the lower half of S, its 2 will be marked.

□

Combining the above and considering Step 3 of the algorithm, we have the following.

Theorem 1 *Any two columns of the matrix B have two pairs of consecutive elements across the two columns, and they satisfy the all-adjacency requirement.*

6 Concluding Remarks

This paper lays the foundation for a new design of shadowed disks. The design favors systems in which the requests are mostly read requests. Such systems include digital libraries and all kinds of information servers and data warehouses, many of which are available over the Internet. We have presented the algorithm that can generate the necessary information, in terms of a perfect all-adjacency matrix, to be used for arranging the logical cylinders in the disk array, and proved its correctness.

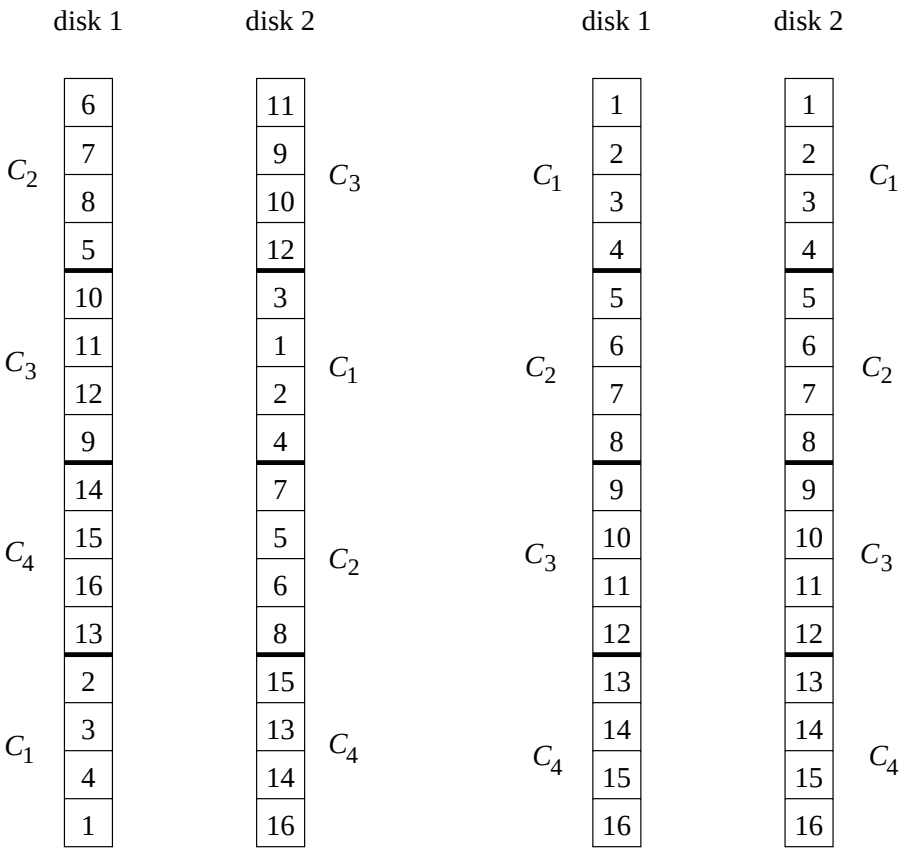
The method requires, for c cylinders, $\frac{c}{2}$ disks, a number which might not be always feasible in practice. To use fewer than $\frac{c}{2}$ disks, we can divide the cylinders into clusters, each consisting of a subset of cylinders, and then the algorithm can be applied first to clusters, and then to cylinders (or another level of clusters) within clusters. We give a "2-level" example here. Suppose $c = 16$ and there are only two disks available. Two disks can support 4 cylinders ($c = \frac{d}{2}$) according to the algorithm. We therefore divide the 16 cylinders into 4 clusters:

$$C_1 = \{1, 2, 3, 4\}$$
$$C_2 = \{5, 6, 7, 8\}$$
$$C_3 = \{9, 10, 11, 12\}$$
$$C_4 = \{13, 14, 15, 16\}$$

The matrix B for $c = 4$ is as follows.

$$B = \begin{bmatrix} 2 & 3 \\ 3 & 1 \\ 4 & 2 \\ 1 & 4 \end{bmatrix}$$

The numbers of the matrix B are applied to laying out the *clusters* in the disks, and then recursively to cylinders within a cluster. In the latter, a 1 in B corresponds to the first element of the cluster, 2 to the second element, and so on. The result is shown in Figure 4(a). In this example system, following a write, the two heads will be positioned at the same cluster in the two disks; to serve the next read operation, the selected head needs to travel at most a distance of two clusters, or 7 cylinders. If the requested cylinder is within the same cluster where the head is positioned, the



(a) new scheme (b) regular scheme

Figure 4: A 16-cylinder, 2-disk system

seek distance is one cylinder. In general, the seek distance for read is bounded from above by $O(\frac{c}{2d})$ for c cylinders and d disks using our method.

Figure 4 also shows a regular shadowed disk system using the same hardware. A comparison between the two schemes gives the following advantages of the new scheme over the regular scheme.

- If either anchoring or confinement is implemented for the new scheme, the seek distance for any read is bounded by two clusters (for anchoring) or three clusters (for confinement). The seek distance for read in the regular scheme can be as long as four clusters.

- Suppose anchoring or confinement is also implemented for the regular scheme such that the two heads are bound to the upper two clusters and the lower two clusters respectively; then the seek distance for read in is reduced to one cluster (for anchoring) and two clusters (for confinement). However, in either case, a write would destroy the anchoring or confinement property. The new scheme does not have this problem. Also, with anchoring in the regular scheme, those cylinders that are closer to the heads are favored (seek distance is shorter); the new scheme is fairer as each new write would potentially move the heads to a different region of the disks.

Further work includes the study of the write operation. In particular, if a read request is performed as early as possible without having to wait until all the write operations triggered by a preceding write request are finished, how much better on average can a write operation be? It would be worthwhile to compare shadowed disks using this method and the popular RAID (redundant arrays of inexpensive disks) systems [3].

References

[1] D. Bitton and J. Gray, "Disk Shadowing", *Proceedings of 14th International Conference on Very Large Databases*, 1988, 331–338.

[2] D. Bitton, "Arm Scheduling in Shadowed Disks", *Proceedings of Spring COMPCON '89*, 1989, 132–136.

[3] G. Gibson and D. Patterson, “RAID: High-Performance, Reliable Secondary Storage”, *ACM Computing Surveys*, Vol. 26, No. 2, 1994, 145–185.

[4] R. King, “Disk Arm Movement in Anticipation of Future Requests”, *ACM Transactions on Computer Systems*, Vol. 8, No. 3, 1990, 214–229.

[5] R.W.-M. Lo and N.S. Matloff, “A Probabilistic Limit on the Virtual Size of Replicated Disk Systems”, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 4, No. 1, February 1992, 99–102.