Logical vs. Physical Disk Shadowing

Goetz Graefe, Leonard D. Shapiro

CU-CS-497-90 November 1990

Department of Computer Science
University of Colorado at Boulder
Campus Box 430
Boulder, Colorado 80309-0430

(303) 492-7514
(303) 492-2844 Fax
graefe@boulder.colorado.edu

Logical vs. Physical Disk Shadowing

Goetz Graefe, Leonard D. Shapiro

Abstract

Since the cost of redundant storage is less important for many applications than the cost of failures and down-times, some database systems offer facilities for maintaining redundant disks. Bitton and Gray analyzed shadowing physical units such as disk pages over multiple devices. Instead, we propose to replicate logical units such as relations and indices. Logical shadowing allows using different physical organizations, e.g., multiple different sort orders and clustering indices. Physical organization and clustering are important determinants in retrieval costs; having multiple clustering schemes increases the number of attributes for which the system achieves optimal performance for range queries and merge joins.

1. Introduction

For many applications, down-times are unacceptable, and database management systems for such applications offer facilities to maintain redundant copies of the database. According to Bitton and Gray, "Disk shadowing is a technique for maintaining a set of two or more identical disk images on separate disk drives. Its primary purpose is to enhance reliability and availability of secondary storage by providing multiple paths to redundant data. However, shadowing can also boost I/O performance." [3] They provide insights how disk shadowing can reduce seek distances for read operations, in principle by using the "shadow set" of "mirrored" disks as one disk with multiple access arms. For write operations, however, shadow disks are more expensive because a new page image has to be written to all disk devices. Bitton and Gray show that disk shadowing improves overall I/O performance if read operations constitute about 55% or more of all I/O operations.

In this paper, we demonstrate how disk shadowing can be further improved if not physical page images but logical data sets are mirrored with different clustering strategies on different disks. Consider, for example, a relation R with three attributes A, B, and C, to be mirrored on two disk drives. Assume R is accessed frequently using range predicates on A or B. If R is shadowed physically, i.e., identical page images are kept on the two disk drives, one has to decide whether to cluster R on A or on B. If, however, R is clustered according to A on the first disk and according to B on the second disk, both types of range queries can be executed very efficiently.

If we assume the same logical data to be stored on all disks, reliability and availability considerations for logical shadowing are the same as for physical shadowing, with the exception of performance considerations during a disk's failure and repair. Furthermore, it is possible to combine logical and physical shadowing, e.g., by keeping two disks with R clustered on A and two disks with R clustered on B.

In the remainder of this paper, we elaborate on this intuition. In the next section, we briefly survey related work. In Section 3, we present more concrete examples and explore how widely applicable logical shadowing is. A performance comparison of logical and physical shadowing is presented in Section 4. We offer our conclusions in Section 5.

2. Related Work

First among the related work is Bitton and Gray's analysis of disk shadowing [3]. Their assumption is that exact mirror images of pages are kept on different devices. The freedom to choose a device makes reads faster; the need to write to all devices makes writes slower. They conclude that if the system performs more reads than writes, overall system performance is improved by disk shadowing. Since writes make disk shadowing expensive, it would be very interesting to analyze the effect of write-only disk caches [21] on the performance of physical shadowing. Write-only disk caches retain dirty pages in safe RAM and attempt to piggy-back writes onto reads to the same cylinder, thus allowing write operations without delay or cost.

Another attempt to reduce I/O costs is disk striping [20] in which pages of a file are distributed over multiple disks. Just as for shadowing, one can distribute pages over disks, or one can distribute records or tuples. Such partitioning schemes were used extensively in the GAMMA database machine [6, 7]. GAMMA employed both a physical partitioning scheme called round-robin-partitioning and two logical partitioning schemes, called range-partitioning and hash-partitioning. Physical partitioning allows faster scans of the entire file; logical partitioning, in addition, can direct a request only a selected set of disks if the search predicate is suitably related to the partitioning attribute.

Partitioning does not improve reliability as replication does; however, special combinations of partitioning and replication have been explored [13]. Replication has also been explored extensively in distributed databases, both for reliability and for proximity of data to users, e.g. in [1, 2, 4, 5, 8, 9, 14, 15, 17-19, 22-26]. The main difference between replication in distributed systems and logical shadowing is that we assume that all disks have the same "distance" from the processors and users, such that transfer times do not force the choice of disks as in distributed database systems. Furthermore, none of these researchers has considered exploiting different ordering and clustering strategies on the various locations.

Finally, since we are concerned here with the costs of using clustered and non-clustered indices, we would like to point out the study by Mackert and Lohman [16] which quantified precisely the cost of using index scans.

3. Applicability of Logical Shadowing

Physical shadowing can be used for any data. Logical shadowing with different clustering strategies can also be used for any dataset, but it only provides advantages if data are accessed in varying ways. Let us consider a simple example, namely the three relations of a simple university database

student (**student-id**, name)
course (**course-no**, title)
enrollment (**student-id**, **course-no**, grade)

with the key attributes printed in bold face. It can be assumed that joins of these three relations will be frequent. For the student and course relations, it makes sense to index and cluster them on their key attributes. For the enrollment relation, one must choose which key attribute to index and cluster on. If one chooses student-id's, joins with students will be very fast, e.g., in a query to list a student's current classes. One the other hand, a join of courses and enrollment will require sorting or hashing, assuming nested loops is not competitive because of the file sizes. For example, to list the class roster for a class requires expensive lookups in the secondary index on course-no in the enrollment relation. If one chooses to cluster enrollment on the course-no, joins with the course relation will be fast and joins with the student relation will be slow.

Joins are one example for which logical shadowing offers performance advantages, but there are other situations in which the limitation to only one clustering choice creates a dilemma. Consider, for example, the customer and account information of a telephone company. The billing department might like to look at accounts in ZIP+4 order to obtain better postage rates for the monthly bill mailings. Directory assistance might want to look at small sections of the customer records, namely all those customers with an equal names. The marketing department might want to target customers in the order of their high monthly long-distance bill, for instance to advertise a special long-distance service. In all these cases, one user group could be served particularly well if the data were clustered according to their selection criterion or required ordering. However, without logical shadowing, the database administrator has to choose the clustering order and, therefore, which department to serve well and which ones to serve only suboptimally.

For the first example above, any other many-to-many relationship would do just as well, e.g., parts and suppliers. For the second example, any dataset with multiple ordered or non-unique attributes that are used as retrieval keys or sort order could be substituted. The problem in all these cases is, obviously, that one can cluster data according to one criterion only. We content, however, that if one keeps multiple copies of a dataset anyways as assumed in disk shadowing, one might as well keep the various copies in different orders and using different clusterings to serve all or almost all users optimally.

There are two disadvantages of maintaining different clusterings. First, maintenance is more expensive since more indices need to be updated for each data update. In particular, if one of the clustering attributes is updated, records have to be moved on that disk. Second, recovery of a single disk is more complex because it requires not only copying from a mirror copy as in physical shadowing but also sorting from another disk into the clustering that was lost in the disk failure.

The first point is a performance tradeoff: it is trivial that the overhead of maintaining complex auxiliary data structures might be larger than their performance advantage if updates are very frequent. We will explore this tradeoff in the next section. For the second point, we offer three possible viewpoints. First, if the remaining disk(s) include secondary indices on the clustering attribute of the failed disk, the entire system should still perform as well as a system without

shadowing or a system using physical shadowing with a failed disk. Second, fast sorting and clustering methods are available — it should be possible to sort and rebuild two indices for a file of 100 MB in about 2 minutes using a parallel machine [11, 12]. Third, it is possible to keep multiple copies of each ordering, i.e., to combine physical shadowing in addition to logical shadowing.

Implementing logical shadowing might seem rather cumbersome; however, many relational database management systems already offer sufficient physical design options to exploit logical shadowing. For example, to shadow a relation R(A,B,C) which is clustered on A on disk 1, one creates a composite (multi-attribute) index on (B,A,C) on disk 2. This index is sorted and clustered on B, and it is maintained automatically when R is updated. Most database vendors have implemented their query optimizers such that they consider scanning the index only. For queries that would benefit from a clustering on B, they recognize that an index scan is sufficient to retrieve all attributes of R and omit the lookup in R from any query evaluation plan. Thus, all pieces necessary to obtain the performance advantages of logical shadowing are already available in many relational database management systems.

For availability and recovery, however, two pieces are missing. First, if disk 1 with R crashes, most database systems will not support recovery of R from the index but will recover from backups and logs. Second, during the disk repair, R is deemed unavailable and the database system does not use the index to satisfy queries against R. Thus, these two special cases should be included in the database software to exploit logical shadowing to its full potential.

4. Performance Analysis

In the following analysis, we compare the advantages and disadvantages of physical and logical shadowing. The analysis uses only approximations because each database system implementation uses different storage structures and cost functions. Therefore, modelling any one database management system precisely would not add to our purpose, which is to demonstrate the validity of logical shadowing as a general performance technique and to shed some light on the tradeoffs between maintenance effort and retrieval speedup.

We only consider two disks, used either to mirror pages of a relation (physical shadowing) or to keep the relation in two different orderings (logical shadowing). Let there be three attributes, two of which are used as keys in retrievals (attributes A and B) and one that is to be retrieved (attribute C). We also assume that an index lookup requires two I/Os (which is probably true for most real B-trees) and that F records fit into each page.

Both retrieval attributes are indexed in both shadowing schemes. In other words, we use one clustering and one non-clustering index in physical shadowing (without loss of generality let attribute A be the clustering one) and one clustering index and one non-clustering index on each disk in logical shadowing (clustering index on A and non-clustering index on B on disk 1, clustering index on B and non-clustering index on A on disk 2). We have chosen these physical designs because they both allow indexed retrievals on A and B before or after disk failure.

Let us consider a query mix that contains R_1 single-record retrievals, R_A range-retrievals of N records in order of attribute A, R_B range-retrievals of N records in order of attribute B, U_B single-record updates which modify attributes B and C, and U_C single-record updates which modify only attribute C. For updates, let us assume that the record to be updated is identified by an attribute value for A. We have chosen these operations because we believe they reflect realistic usage patterns. Let us consider their costs.

The single-records retrieval can be performed equally fast using a clustered and a non-clustered index; thus, we can decide which logical shadow to use depending on current load on the disks. The difference to physical shadowing is that this decision is fixed for all reads in the index and the data file for an entire retrieval; in physical shadowing, each I/O access can be done on either disk. Retrieving N records in clustering order requires an index lookup plus retrieving about N/F pages with records. In non-clustering order, N pages are required in addition to the index lookup.

For updates, if only attribute C is modified, the cost in either shadowing scheme is one index lookup on A to determine the record to be updated and the record modifications on both disks. If attribute B is also updated, the cost remains the same for physical shadowing and for disk 1 in logical shadowing. On disk 2, the updated record must be moved to a different page to

maintain clustering on B. The cost on disk 2, therefore, is one index lookup in the index on A to determine the record to be updated, two more index lookups for entry removal and insertion on the index on B, two writes to save the updated leaf pages, plus two read-write pairs to move the data record to a new location.

Table 4.1 shows the cost formulas for the considered operations. The cost values for single-page read and write were taken from Table 1 in [3] and express the expected seek time relative to the disk's data band. To obtain absolute seek times, they must be divided over the disk's seek speed. Even as they are, they can be compared as if they were seek times and therefore serve the purpose of comparing the relative performance of the two shadowing schemes. The last formula in Table 4.1 considers only the second disk on which the modified record has to be moved since operations on disks 1 and 2 can proceed in parallel and the cost on disk 2 surpasses that on disk 1. The formulas for retrieval on B contains the reason for using logical rather than physical shadowing, i.e., the high cost of sequential scans of secondary indices.

Operation	Frequency	Cost in Physical Shadowing	Cost in Logical Shadowing
Page read	Rd	0.16	0.28
Page write	Wr	0.43	0.28
Single-record retrieval	R_1	$(2+1)\,Rd$	$(2+1)\,Rd$
N-record retrieval on A	R_A	$(2+N/F)\,Rd$	$(2+N/F)\,Rd$
N-record retrieval on B	R_B	$(2+N)\,Rd$	$(2+N/F)\,Rd$
Single-record update of C only	U_C	$2Rd + (Rd+Wr)$	$2Rd + (Rd+Wr)$
Single-record update of B and C	U_B	$2Rd + (Rd+Wr)$	$3\times 2Rd + 2Wr + 2\,(Rd+Wr)$

Table 4.1. Costs of Operations.

Parameter		Value
R_1	Single-record retrievals	100
R_A	Range retrievals on A	50
R_B	Range retrievals on B	0–50
U_B	Updates of B and C	0 or 50
U_C	Updates of C only	50
F	Records per page	10
N	Records per range retrieval	10 or 50

Table 4.2. Parameter Values in Cost Comparisons.

The following figures show the estimated costs of the operation mix using the parameter values given in Table 4.2. Using the costs from Bitton and Gray [3], the figures indicate the relative seek costs of physical shadowing (solid lines) and logical shadowing (dashed lines) for increasing values of R_B, the number of range retrievals on attribute B. The four graphs show the costs for different values of U_B and N, the number of updates of attribute B and the number of records in a range retrieval.

Figure 4.1 shows the situation if there are no updates of B and a relatively small N. The costs of the two schemes are almost equal over the entire range. Physical shadowing is a little less expensive for R_B=0 reflecting the smaller seek cost of physical shadowing identified by Bitton and Gray. However, the curve for physical shadowing is steeper, indicating the larger incremental cost for each range retrieval on B using a secondary index scan.

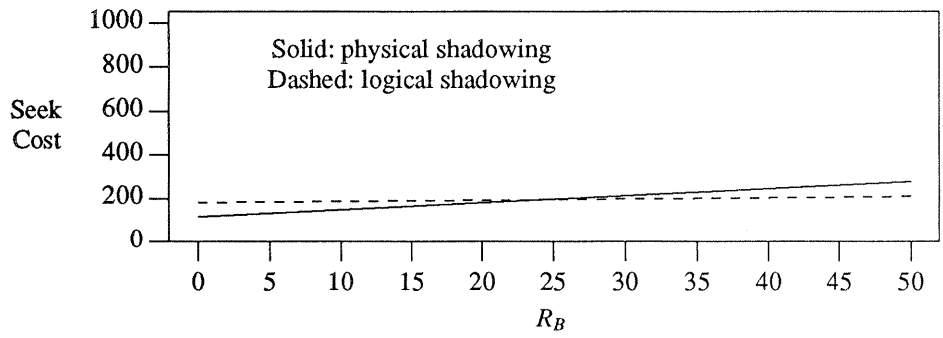

Figure 4.1. Cost Comparisons for $U_B = 0$, N = 10.

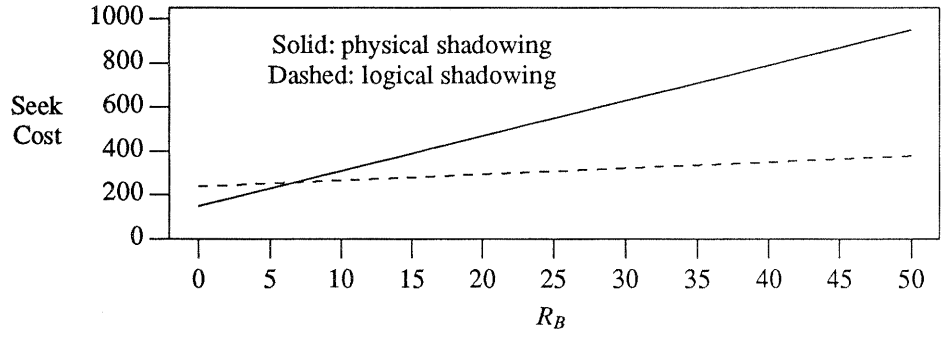

Figure 4.2. Cost Comparisons for $U_B = 0$, N = 50.

Figure 4.2 shows the same situation with a larger N, i.e., more records per range retrieval. The costs for R_B=0 are higher than in the previous figure. for both shadowing schemes, reflecting the higher retrieval costs for range retrievals on A. The slopes of the curves are more interesting, however. While the incremental cost for each range retrieval on B in logical shadowing is quite small, it is rather larger for physical shadowing. This graph shows the purpose of logical shadowing: if there are many multiple-record retrievals on more than one attribute (A and B here), providing two sort and clustering orders gives a significant performance advantage.

Figure 4.3 shows the additional cost of logical shadowing. If updates to one of the clustering attributes (B here) are frequent, logical shadowing incurs an additional cost over physical shadowing. If the performance advantage of logical clustering is small, either because there are few range retrievals on the second attribute or because the number of records per retrieval is small, logical shadowing is inferior to physical shadowing.

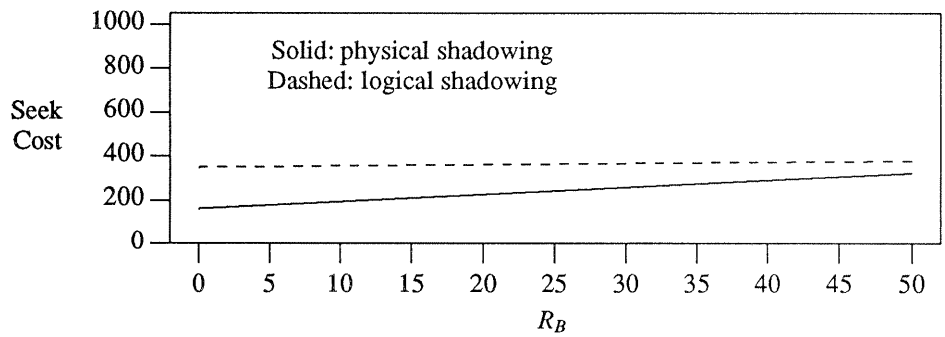

Figure 4.3. Cost Comparisons for U_B = 50, N = 10.

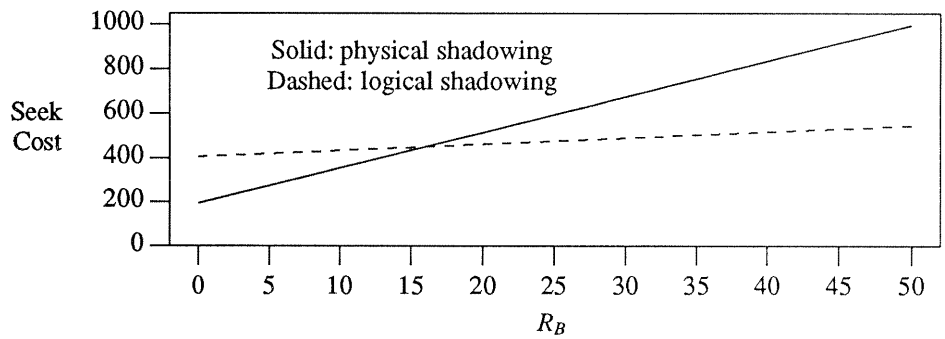

Figure 4.4. Cost Comparisons for U_B = 50, N = 50.

Figure 4.4 puts the advantages and disadvantages of logical shadowing observed in the previous graphs together and reflects their tradeoff. For few range retrievals on B, physical shadowing is faster because of smaller seek times and less update overhead. For larger range retrievals on more than one attribute, however, logical shadowing offers a significant performance improvement, even when the additional maintenance cost is taken into account.

5. Summary and Conclusions

In this paper, we have introduced a new concept, called "logical disk shadowing." It replaces replication of physical page images as used in what we call physical disk shadowing [3] with replication of logical data sets in different sort order, clusterings, etc. We demonstrated that realistic situations exist in which logical shadowing outperforms physical shadowing by a significant margin.

Our analysis only covered the simplest case with two disks and two interesting sort orders. This analysis could be carried further to consider more disks and clustering attributes. Since logical and physical shadowing can be combined, such an analysis should take combinations into consideration.

The performance can also be compared by relating hardware investment and resulting performance. Most disk drives and systems can deliver about 10 times more pages in sequential I/O than in random I/O. Furthermore, for random reads as used in unclustered index scans, 1 of maybe 10 records in each page is needed. Multiplying these two factors of 10 indicates that 100 disk drives performing random reads of individual records deliver no more records than 1 disk drive performing a clustered scan. Thus, if records are needed in more than one ordering, it is less expensive to invest in disk space for a logical shadow than in more disk arms to perform faster secondary index scans.

It is interesting to note that if the enhanced reliability and availability of shadowing is not required for an application, logical shadowing or replication can be implemented on a single disk. The only additional consideration is that multiple concurrent scans of clustered indices on a single disk can increase disk seeks to almost the level of unclustered index scans.

The clustering schemes considered here, namely sort order clustering for a relation, are not the only ones to which logical shadowing applies. Other schemes include master-detail clustering in relational systems as well as the wealth of possible clustering schemes in object-oriented systems. We plan on exploring the effect of logical shadowing in object-oriented systems as part of the REVELATION OODBMS project [10].

Acknowledgements

This research has been partially supported by the Oregon Advanced Computing Institute (OACIS) and by NSF awards IRI-8805200 and IRI-8912618.

References

1. K. Abbott and D. McCarthy, "Administration and Autonomy in a Replication-Transparent Distributed DBMS", *Proceedings of the Conference on Very Large Databases*, Long Beach, CA., August 1988, 195-205.
2. P. A. Bernstein and N. Goodman, "An Algorithm for Concurrency Control and Recovery in Replicated Distributed Databases", *ACM Transactions on Database Systems 9*, 4 (December 1984), 596-615.
3. D. Bitton and J. Gray, "Disk Shadowing", *Proceedings of the Conference on Very Large Databases*, Long Beach, CA., August 1988, 331-338.
4. M. Carey and M. Livny, "Distributed Concurrency Control Performance: A Study of Algorithms, Distribution, and Replication", *Proceedings of the Conference on Very Large Databases*, Long Beach, CA., August 1988, 13-25.
5. B. Ciciani, D. M. Dias and P. S. Yu, "Analysis of Replication in Distributed Database Systems", *IEEE Transactions on Knowledge and Data Engineering 2*, 2 (June 1990), 247-261.
6. D. J. DeWitt, R. H. Gerber, G. Graefe, M. L. Heytens, K. B. Kumar and M. Muralikrishna, "GAMMA - A High Performance Dataflow Database Machine", *Proceedings of the Conference on Very Large Data Bases*, Kyoto, Japan, August 1986, 228-237.
7. D. J. DeWitt, S. Ghandeharadizeh, D. Schneider, A. Bricker, H. I. Hsiao and R. Rasmussen, "The Gamma Database Machine Project", *IEEE Transactions on Knowledge and Data Engineering 2*, 1 (March 1990), 44-62.
8. H. Garcia-Molina, "Performance of Update Algorithms for Replicated Data in a Distributed Database", *Ph.D. Thesis*, Palo Alto, CA, June 1979.
9. H. M. Gladney, "Data Replicas in Distributed Information Services", *ACM Transaction on Database Systems 14*, 1 (March 1989), 75-97.
10. G. Graefe and D. Maier, "Query Optimization in Object-Oriented Database Systems: A Prospectus", in *Advances in Object-Oriented Database Systems*, vol. 334 , K. R. Dittrich (editor), Springer-Verlag, September 1988, 358-363.
11. G. Graefe and S. S. Thakkar, "Tuning a Parallel Database Algorithm on a Shared-Memory Multiprocessor", *submitted for publication, also CU Boulder Comp. Sci. Tech. Rep. 470*, April 1990.

12. G. Graefe, "Parallel External Sorting in Volcano", *submitted for publication, also CU Boulder Comp. Sci. Tech. Rep. 459*, February 1990.

13. H. I. Hsiao and D. J. DeWitt, "Chained Declustering: A New Availability Strategy for Multiprocessor Database Machines", *Proceedings of the IEEE Conference on Data Engineering*, Los Angelos, CA, February 1990, 456.

14. T. A. Joseph and K. P. Birman, "Low Cost Management of Replicated Data in Fault-Tolerant Distributed Systems", *ACM Transcations on Computer Systems 4*, 1 (February 1986), 54.

15. G. Lohman, C. Mohan, L. Haas, D. Daniels, B. Lindsay, P. Selinger and P. Wilms, "Query Processing in R*", in *Query Processing in Database Systems*, W. K. D. S. R. D. S. Batory (editor), Springer, Berlin, 1985, 31-47.

16. L. F. Mackert and G. M. Lohman, "Index Scans Using a Finite LRU Buffer: A Validated I/O Model", *ACM Transaction on Database Systems 14*, 3 (September 1989), 401.

17. A. Milo and O. Wolfson, "Placement of Replicated Items in Distributed Databases", *Lecture Notes in Computer Science 303* (April 1988), 414, Springer Verlag.

18. R. Mukkamala, S. C. Bruell and R. K. Shultz, "A Heuristic Algorithm for Determining a Near-Optimal Set of Nodes To Access in a Partially Replicated Distributed Database System", *Proceedings of the IEEE Conference on Data Engineering*, Los Angeles, CA., February 1988, 330-336.

19. R. Mukkamala, "Measuring the Effect of Data Distribution and Replication Models on Performance Evaluation of Distributed Database Systems", *Proceedings of the IEEE Conference on Data Engineering*, Los Angelos, CA, February 1989, 513.

20. K. Salem and H. Garcia-Molina, "Disk Striping", *Proceedings of the IEEE Conference on Data Engineering*, Los Angeles, CA., February 1986, 336.

21. J. A. Solworth and C. U. Orji, "Write-Only Disk Caches", *Proceedings of the ACM SIGMOD Conference*, Atlantic City, NJ., May 1990, 123.

22. S. H. Son, "Synchronization of replicated data in distributed systems", *Information Systems 12*, 2 (1987), 191.

23. M. Stonebraker and G. A. Schloss, "Distributed RAID — A New Multiple Copy Algorithm", *Proceedings of the IEEE Conference on Data Engineering*, Los Angelos, CA, February 1990, 430.

24. J. Tang and N. Natarajan, "A Scheme for Maintaining Consistency and Availability of Replicated Files in a Partitioned Distributed System", *Proceedings of the IEEE Conference on Data Engineering*, Los Angelos, CA, February 1989, 530.

25. I. L. Traiger, J. Gray, C. A. Galtieri and B. G. Lindsay, "Transactions and Consistency in Distributed Database Systems", *ACM Transactions on Database Systems 7*, 3 (September 1982), 323-342.

26. E. Wong and R. H. Katz, "Distributing a Database for Parallelism", *Proceedings of the ACM SIGMOD Conference*, San Jose, CA., May 1983, 23-29.