WHAT IF MASS STORAGE WERE FREE?

George Copeland
Tektronix, Inc.
Beaverton, Oregon 97077

Abstract

This paper investigates how database systems would be designed and used under the limiting-case assumption that mass storage is free. It is argued that free mass storage would free database systems from the limitations and problems caused by conventional deletion techniques. A non-deletion strategy would significantly simplify database systems and their operation, as well as increase their functionality and availability. Consideration of this limiting case helps shed light on a more realistic argument: if the cost of mass storage were low enough, then deletion would become undesirable.

It is also argued that the often labor-intensive costs and time delays involved in archival and retrieval of older data can be minimized if a single technology were available with low-cost on-line storage and a low-cost archival media with long shelf life.

Optical discs promise to come one to two orders of magnitude closer to the limiting case of free mass storage than ever before. Other features of optical discs include improved reliability and a single technology for both on-line and archival storage with a long shelf life. Because of these features and because of (not in spite of) their non-deletion limitation, it is argued that optical discs fit the requirements of database systems better than magnetic discs and tapes.

1 Introduction

The cost of computer hardware has been decreasing at a drastic yet consistent rate for at least three decades. In the past, system designers have had to compromise functionality, reliability, and other human-oriented costs in order to achieve hardware efficiency. As hardware costs decrease by orders of magnitude, however, thresholds are reached at which these compromises must be relaxed in order to minimize total cost. Examples of this situation are the trend from assembler languages toward higher-level languages, the trend from large monolithic programs with unrestricted use of the GO TO toward structured programs, and the trend from all-purpose, primitive human interfaces toward application-oriented ones. In each of these examples, significant changes in system concepts have been required to place a greater emphasis on human costs than on hardware costs.

This paper considers the limiting-case assumption that mass storage is free. Its purpose is to examine some of the implications that this assumption would have concerning the design of future database systems.

Section 2 argues that this free-storage assumption leads to the elimination of deletion in database systems. A non-deletion strategy is suggested using timestamps that would allow significant simplifications in database systems and their operation, as well as a significant increase in application functionality and data integrity. Also, the non-deletion strategy eliminates the need to make periodic checkpoint rollouts for recovery from errors and its resulting impact on system availability. Consideration of this limiting case helps make clearer the more realistic argument that if the cost of mass storage is low enough, then deletion becomes undesirable.

Section 3 addresses current problems in archival procedures. It is argued that the labor-intensive costs and time delays involved in archival and retrieval of older data could be minimized if a single technology were available that had low-cost on-line storage and a low-cost archival media with long shelf life.

Section 4 describes how optical discs promise a single technology with one to two orders of magnitude lower cost and volume for on-line and archival storage than the conventional combination of magnetic discs and tapes. Improvements in physical reliability are described. In addition, their "limitation" of non-deletion improves the integrity of data. Because of these properties, it is argued that optical discs fit database system requirements better than magnetic discs and tapes.

Section 5 summarizes the arguments and states their conclusions.

2 Deletion considered harmful

This section argues that significant improvements in functionality, integrity, availability, and simplicity can be achieved in database systems if the deletion mechanism is

eliminated. Section 2.1 describes a non-deletion strategy that uses timestamps to identify which values are current. Section 2.2 describes how such a non-deletion strategy would significantly improve functionality by allowing access to past as well as current information in a simple and uniform way. Section 2.3 describes the advantages of non-deletion using timestamps over periodic checkpoints for recovery from errors. Section 2.4 describes how timestamps can also provide a method for logical sychronization of a network of distributed databases, while reducing the need for lockout. Section 2.5 discusses the impact of non-deletion on retrieval performance.

2.1 A non-deletion strategy using timestamps

The deletion function is not needed in database systems. This section outlines a non-deletion strategy that is consistent with current concepts in the database field.

Database systems deal with the inclusion, modification, removal, and retrieval of descriptive data about real-world objects. For example, an individual may be included in an employee file when hired. While the individual is employed, his descriptive data may undergo several modifications. The individual will eventually be removed from the active employee file when employment is terminated. Descriptive data may be retrieved either during or after employment.

Because any or all of an object's descriptive data may be modified, identification of the object may become difficult. Smith and Smith's (1978) principle of individual preservation argues the importance of preserving the individual identity of an object. Even though any descriptive data is modified, an object is the same and should be identifiable as such. This principle is important for maintaining relationships that an object may have with other objects.

Keys, such as social security numbers, were proposed by Codd (1970) to accomplish this unique identification. This technique prohibits the modification of keys. However, since keys also serve as descriptive data, their values may occasionally need modification. For example, a social security number may be entered incorrectly when the individual was hired.

More recently, several workers (Abrial 1974, Hall et al. 1976, Kent 1978, Codd 1979) have suggested using surrogate values that are created by the database system to serve as unique and invariant identifiers. By separating the identifier from descriptive data, surrogates simplify the task of preserving individual identity, since users have no need to modify them. Surrogates eliminate the need for keys, so that all descriptive data can be modified by users in a uniform manner without risking loss of individual identity. The strategy suggested in this section uses surrogates.

To support a non-deletion strategy, another mechanism called timestamps is needed. When modifying the description of an object without deletion, it becomes difficult to identify the current version of the descriptive data. A sequence number would suffice to solve this problem. However, the value of the time when the modification was made serves this purpose as well as others to be described later. This value of the transaction clock time is called here a timestamp. The following paragraph describes how surrogates and timestamps are used to accomplish inclusion, modification, removal, and retrieval without using deletion.

An object is included in the model by inserting an entry with a new, unique, system-created surrogate, with a current timestamp, and with any descriptive data that is known. An object is modified by inserting an entry with the individual's original surrogate, with a new current timestamp, and with the modified descriptive data. An object is removed by inserting an entry with the individual's original surrogate, with a new current timestamp, and with an indication of removal. Both modification and removal would normally first require a retrieval to obtain the object's original surrogate. A retrieval of current information is accomplished by searching for the most recent timestamp. A retrieval of past information is accomplished by searching for the timestamp that is most recent as of some specified time in the past.

2.2 Improved functionality

A database is a model of some real-world system. To maximize the functionality of the database, the model should be as close as possible to the underlying real-world system. This section argues that a non-deletion strategy can significantly improve the functionality of database systems by allowing the model and usage of data to be closer to real people and their systems.

2.2.1 The importance of access to past states

In human memory, no deletion mechanism exists (Underwood 1969, Nielsen 1958). Although human memory exhibits a decay characteristic, people do not delete. The deletion concept was invented to reuse expensive computer storage.

In the real world of everyday life, people commonly use knowledge of past information to make decisions that control their individual lives, their governments, their businesses, and other organizations. For example, it is quite common to make comparisons of current data with previous periods for trend analysis. Auditing is commonly used to insure the internal control of almost every organization. Also, people are often interested in how up-to-date information is, to determine whether it is currently valid. A historical perspective has often shed light on the analysis of a current problem. Information about an object may be retrieved based on past information. For example, one can retrieve the employees who worked in the Toy department in May of 1974 and currently work in the Hardware department. Many other uses of past information are quite common.

The following control structure is commonly used in programming languages and in everyday life: if this is true, then..., else..., end if. A similar control structure using past tense is

commonly used in everyday life: if this was true, then..., else..., end if. Harvill (1978a, 1978b) has suggested that such a past-tense control structure be used in programming languages. Harvill has argued that the use of past tense offers many psychological advantages, since it allows programs to include algorithms that are commonly used by people.

Access to past states of data has rarely been used in computer systems because the cost of computer-readable storage has been too high. Section 4 describes how optical discs promise to provide a low-cost storage, allowing the use of past tense to be more practical.

2.2.2 Most databases require a non-deletion policy

Most existing databases (not necessarily computer automated) that directly model real-world systems require a non-deletion policy. The reason is that access to past states of the data is an inherent part of the application.

For example, a fundamental principle of accounting theory is that no journal entry or ledger posting deletes any information. When an item is purchased on credit, entries are made to both sales (a credit) and accounts receivable (a debit). When the account is paid, entries are made to cash (a debit) and accounts receivable (a credit). The first entry to accounts receivable is not deleted by the second entry. Even entries made by mistake are not deleted. Instead, they are reversed by making an additional entry. Also, the non-deletion policy in accounting is vital to the auditing process required to insure internal control.

All accounting, financial, and legal databases require a non-deletion policy. Most text and document databases require non-deletion. The concept of generations of data is used by many applications. In engineering design and documentation, a new version of the design of an aircraft wing or an LSI layout does not eliminate the previous version. Previous design versions are kept for backup in case the new design turns out to be a step in the wrong direction, for reviewing the progress of the design process, or for evidence of early design work and progress for patent applications. In cartography and census databases, older generations are kept for historical reference. Many other examples exist. The simple non-deletion strategy suggested here directly facilitates databases that have such a required policy.

Many of these real-world databases have not been computer automated because the cost of computer-readable storage has been too high. Instead, paper or microfilm has been used as storage. Section 4 describes how optical discs promise to provide a sufficiently inexpensive storage to allow many of these non-deletion databases to be computer automated.

2.3 Recovery from errors

Computer system reliability (integrity and availability) is an increasingly critical problem. Database systems that are shared among several applications are becoming increasingly important to overall system reliability. This section argues that a non-deletion strategy can significantly improve both integrity and availability.

2.3.1 The importance of reliability in database systems

The reliability of existing systems is unsatisfactory. History suggests that as we move further into the future, more of our real-world systems will be automated with computers. This development is likely to cause computer systems to become more complex and thus less reliable. Also, we will depend more on these computer systems than ever before, so that their reliability will need to increase even further. Because of these two effects, the problem of system reliability (integrity and availability) is becoming increasingly critical.

We can combat this problem with two techniques. Improvements in technological reliability have been and will continue to be a major technique. Secondly, the structured approach of distributed processing promises to allow applications to become more independent and thus easier to control. Application hardware, software, and human interfaces can be tailored to fit the application for improved technological and human efficiency and reliability. Also, users are freed from much of the error-prone bureaucratic cooperation required by large, shared systems. However, even if this structured approach is taken to its limit (no hardware is shared), applications will still have to cooperate because of shared data.

Database system reliability becomes most critical because all applications depend on it. In particular, editing of shared data poses a potential threat to all applications. Data integrity can be improved by having a means for recovery from various types of errors. The mass storage device itself may lose data. This loss can be avoided if the mass storage is reliable, or recovered from if data is stored redundantly on multiple devices. Redundant storage on multiple devices becomes more feasible when the cost of storage is cheaper and is most effective if the storage devices are geographically separated. Several types of editing errors can also compromise data integrity. For example, data can be lost if undesirable deletions are made by errors in the hardware or software of the database or communication system. Even if these systems are ideal, application users can make deletions either interactively or in application programs that later turn out to be wrong. Deletion errors can be avoided if no deletion mechanism exists, or recovered from if periodic checkpoint copies of the data are made.

2.3.2 The periodic checkpoint strategy

Traditionally, the mechanism used for recovery from deletion errors is periodic checkpoint rollout and the maintenance of an audit trail of edits. This mechanism could be considered a form of non-deletion. Users are allowed to pretend they

are deleting. However, the database system actually saves all data and edits, and merely simulates deletion. Deletion errors can be recovered by rollin of the most recent valid checkpoint copy and executing the audit trail of edits, excluding the erroneous edit. Since errors may not be detected immediately, the system must keep rollout copies and audit trails back as far as recovery from an error is needed. This periodic rollout approach has an undesirable impact on system throughput and system availability.

One major problem is its impact on database system throughput. The percentage of system time dedicated to rollout grows as the interval between rollouts is shortened and as the size of the database grows. The percentage of system time dedicated to recovery grows as the interval between rollouts is lengthened and as usage increases because the audit trail of edits grows longer. The interval can be chosen to optimize system throughput (Chandy et al. 1975, Chandy 1975). However, even if this optimization is done, the periodic checkpoint approach can have a major impact on throughput if error rates are high. Since both database size and usage are expected to increase as computer automation increases, the impact of the periodic checkpoint strategy on system throughput is increasingly important.

Another major problem concerns the maximum allowable time that the database system can be unavailable to users. Rollout and recovery each require the database system to become unavailable for editing. Recovery causes unavailability for retrieval. As we move further into the future, databases will grow in size, usage, and importance. This growth in size and usage will increase the number and length of unavailable time periods caused by rollout and recovery. Higher usage and importance will also demand shorter and fewer unavailable periods. Thus, the impact on system availability due to the periodic checkpoint strategy is becoming increasingly critical. Efforts toward allowing database editing during rollout (Rosenkrantz 1978) have led to complex procedures and require complex directories, which are themselves databases that are subject to errors. These factors tend to reduce the reliability of the recovery process itself.

2.3.3 Advantages of non-deletion using timestamps

As mentioned previously, the periodic checkpoint strategy can be considered a form of non-deletion. The rollout copies and audit trails of edits contain all states of the database. For this reason, it is not clear whether the periodic checkpoint or timestamp strategy requires more total (on-line and archival) storage, especially if the great deal of redundancy involved in the checkpoint strategy is considered.

The timestamp strategy avoids the long time periods when the database system is unavailable by eliminating rollout and rollin copying. This is possible because all states of the database are kept intact. Recovery can be accomplished by using the older timestamped data to backup to prior valid states of the data. The timestamp strategy shifts the problem from the burden of checkpointing to the archival of older data. However, since editing does not affect older data under the timestamp strategy, archival of older data can be done concurrently or interleaved with editing. Thus, the timestamp strategy avoids the long periods of system unavailability. Section 3 describes how archival copying can be greatly reduced or eliminated if a single technology is used for both on-line and archival storage. Thus, the timestamp strategy can avoid the system time required for copying, increasing system throughput.

2.4 Timestamps and distributed databases

It is often desirable to distribute data among several database systems, either widely separated or physically adjacent. Distributing data could be desirable for one or more of the following three reasons. One major reason is to minimize communications by exploiting locality of usage. The cost and time delays involved in communication can be minimized if data is clustered physically close to where it is most often used. A second reason is to improve data reliability by redundantly storing data in multiple systems. For example, if every piece of data is stored on two systems, then loss of a single system loses no data. A third reason is to balance the transaction load to remain within the throughput limitations of each database system and the communication system.

A major synchronization problem occurs when an update is made that involves distributed data. Inconsistent data may result if retrieval and editing transactions are not synchronized. Yet accurate physical synchronization is quite difficult between distributed systems, expecially if communication delays are variable. Several strategies have been suggested (Kaneko et al. 1979, Thomas 1978, Reed 1979) that use timestamps that are stored with the data to provide logical synchronization. Stored timestamps can also eliminate the need to lock-out queries from data that is being edited. Instead, a query can be processed concurrently or interleaved with an edit by using data that was current just prior to the beginning of the edit. This advantage applies within a single database system, as well as to a network of database systems.

In light of the other utilities of stored timestamps described in this paper, perhaps another step can be made toward system simplicity and efficiency by accomplishing several functions with this single mechanism.

2.5 Retrieval peformance

A complete analysis of the impact of the timestamp non-deletion strategy on retrieval time would be quite involved. Numerous storage structures and access methods have been used or proposed for the deletion strategy. Each of these combinations would require individual analysis of their compatibility with non-deletion. Furthermore, many new techniques should be investigated with the unique properties of non-deletion in mind from the beginning, rather than simply trying to force deletion storage and access techniques to fit the non-deletion strategy. Such a complete or generalized analysis is beyond

the scope of this workshop paper. Instead, one purpose of this paper is to stimulate research in this area. The following paragraph decribes one tradeoff that seems common to many non-deletion storage strategies.

Let us assume a record-oriented system, where fields are variable in length and separated by tagged delimiters. For example, an employee file might use delimiters such as:

(a)surrogate(b)timestamp(c)name(d)SS#...
(n)department...

Original individual instances might be:

(a)571(b)442(c)Smith(d)410-78-8408...
(n)Toy...

or

(a)622(b)476(c)Sommers(d)512-11-1339...
(n)Cosmetics...

Let us further assume that a modification is needed whereby Smith is transferred to the Hardware department. A tradeoff exists between storage space and retrieval time. An extreme strategy for minimum storage would accomplish the modification by inserting only the modified descriptive data:

(a)571(b)583(n)Hardware

This approach would increase retrieval time because many entries would have to be scanned to find all current information about Smith. An extreme strategy for minimum retrieval time would accomplish the modification by duplicating the rest of the record:

(a)571(b)583(c)Smith(d)410-78-8408....
(n)Hardware...

This approach would require no additional retrieval time to find current information because scanning is avoided. A compromise strategy lies along the continuum between these two extremes, allowing a tradeoff between storage space and retrieval time. Only modified data would be inserted for up to some number N modifications. On each multiple of N modifications, all current information would be gathered and inserted. This approach guarantees a maximum number of entries that must be scanned to retrieve current information, placing an upper bound on additional retrieval time. The value of N would be determined by a tradeoff between the cost of storage and the economic importance of retrieval time.

It should be mentioned that any decrease in retrieval performance due to non-deletion must be considered in light of the improvements in system throughput and availability due to non-deletion described in this paper.

3 Archival

This section describes several techniques for reducing the labor-intensive costs and time delays caused by archival.

3.1 Reduced on-line cost

If mass storage were free, archival would not be necessary because all data would be stored on-line. More realistically, on-line mass storage will always have a finite cost. However, if mass storage were considerably less expensive, archival would be considerably reduced because more of the data would be stored on-line. Thus, a major technique for reducing the costs and time delays caused by archival is to reduce the cost of on-line mass storage.

3.2 A single technology for on-line and archival

Currently, most larger computer-automated databases use two separate technologies for on-line and archival storage -- magnetic discs and tapes. This two-technology approach requires copying between the two for both archival and retrieval of older data. This copying can occupy a significant amount of system resources and can cause significant time delays.

However, if a single technology is used for both on-line and archival storage, then the following strategy can reduce or eliminate copying. The strategy would require using multiple (two or more) drives and would require data to be stored chronologically according to media. As an example, suppose that two drives called A and B were used. Data insertions would be stored on one drive, say A containing media M1, until filled. Then drive B containing media M2 would be used. Before M2 is filled, media M1 on drive A would be archived by replacing it with an empty media M3. When media M2 is filled, M3 on drive A would be used. This circular strategy would require only one drive at a time to become unavailable for a short period. Furthermore, it would be the drive containing the oldest data.

One problem with this circular strategy is that some current-yet-old data will be archived, requiring physical mounting and removal of media for retrieval. Alternatively, all current data could be copied from the media being replaced prior to replacement. This compromise strategy would allow access to all current data with no human activity and shorter time delays at the expense of some copying time. It should be stressed again that this type of copying does not require the system to become unavailable, since non-deletion using timestamps does not allow editing of older data.

3.3 Mechanical archival

Another technique for eliminating the human activity and reducing the time delays for archival and retrieval of older data is to use a mechanical means of mounting and removing the archival media. One example of this "juke-box" technique is the IBM 3850 Mass Storage System, where magnetic tapes are mechanically mounted and removed. Large magnetic discs are used as a buffer or staging device. This device suffers from significant mechanical reliability problems and high cost. Also, since two technologies are used, it suffers from the previously described problems of copying between media.

A "Lazy-Susan" approach has been suggested for optical discs (Kenny et al. 1979), where discs are archived mechanically. Since this approach uses a single technology, no copying is required. Although this approach promises to be simpler and considerably less expensive than the IBM 3850, its mechanical reliability is unknown.

4 The technological promise of optical discs

Currently, most larger computer-automated databases use magnetic discs for on-line storage and magnetic tapes for archival. Magnetic disc media is usually either too expensive or is non-removable and therefore cannot be used for archival. Magnetic tape is too expensive and too slow for on-line storage. Thus two technologies are needed.

Optical discs are a non-deletion mass storage technology. Section 4.1 describes its advantages over magnetic discs for on-line storage. Section 4.2 describes advantages of optical discs over magnetic tapes for archival storage.

4.1 On-line storage

Optical discs (Bulthuis et al. 1979, Laub 1980) promise to offer cheaper cost per bit than magnetic discs by one to two orders of magnitude. These advantages would allow the on-line storage for a database system to be increased significantly. Read and write times seem roughly equivalent for both optical and magnetic discs.

Optical discs also promise several advantages over magnetic discs concerning reliability. Since the laser read head is much further away from the media, the possibility of a head crash and the need for mechanical contact are virtually eliminated. Secondly, optical discs promise to have significantly fewer problems with mechanical vibration. Thirdly, since the laser read head focuses within rather than on the surface of the transparent plastic or glass media, problems with dust, smoke and scratches are significantly reduced. Also, problems with loss of data due to high temperatures are reduced, since optical disc storage is more permanent, and the glass optical disc media are less destructible by fire. Optical disc drives, however, will require periodic maintenance due to the limited lifetime of the solid-state laser read-write head.

4.2 Archival storage

The optical disc media (Kenney et al. 1979) promises a lower cost per bit than magnetic tape by roughly one order of magnitude. Also, the optical disc media requires a smaller volume per bit than magnetic tape by one to two orders of magnitude. These advantages significantly reduce the human activity involved in mounting and removing the media.

The optical disc media also promises significantly reduced maintenance costs. Practical magnetic tape lifetime is roughly two years. Optical disc media promises a greater-than-ten-year lifetime.

5 Summary and conclusions

The high cost of mass storage has caused system designers to compromise database functionality, availability, and simplicity to achieve hardware efficiency. As this cost is reduced, these compromises can begin to be relaxed. This situation is analogous to the move away from assember languages toward higher-level languages, and to the move away from large monolithic programs with unrestricted use of the GO TO toward structured programs. In each case, the relative shift in hardware and human costs has caused significant system-level changes in order to minimize total cost. This paper has examined the limiting case of free mass storage in order to shed light on how future database systems should be designed. The major conclusion is that optical discs are more consistent with future database systems than magnetic discs and tapes. The following paragraphs summarize the arguments that support this conslusion.

Section 2 argued that the deletion function was introduced into computer systems as a compromise that favors storage costs over other costs. As the cost of mass storage is reduced, however, these other costs become more significant. Elimination of the deletion function can improve database systems in several ways. First, access to past information is directly supported, so that past tense can be used in applications. Secondly, databases requiring a non-deletion policy can be directly supported. Thirdly, recovery from errors can be achieved with a minimum of system unavailability and system resources. Fourthly, stored timestamps offer a method of achieving logical synchronization among a distributed network of databases. Furthermore, these advantages are achieved using a single mechanism of stored timestamps, so that system efficiency and simplicity is maximized.

Optical discs facilitate a non-deletion policy because they offer significantly lower cost for on-line storage than magnetic discs. Their technological limitation of no deletion mechanism becomes an advantage. In fact, optical disc systems could be further improved if the system could guarantee no rewriting of a sector once it has been written. Furthermore, optical discs offer improved technological reliability over magnetic discs.

Section 3 described how the labor-intensive costs and time delays of archival could be minimized if a single technology were available that had low-cost on-line storage and a low-cost archival media with long shelf life. The low on-line storage cost of optical discs reduces the need for archival. Optical discs offer a single technology for both on-line and archival storage, so that time-consuming copying can be minimized for archival and retrieval of older data. Optical discs offer a higher-density media, so that fewer media need be mounted and removed. Optical-disc media offer a longer shelf life than magnetic tape, so that less maintenance is required for archival media. The technological limitation of no deletion mechanism of optical discs is ideal for archival since it improves data integrity and since archival

data is not modified.

Acknowledgments

Thanks to Jack Grimes for his suggestion of how to organize the argument presented in this paper.

References

J.R. Abrial, "Data Semantics", in Data Base Management, edited by J.W. Klimbie and K.L. Koffeman, North-Holland Pub. Co., Amsterdam, New York, Oxford (1974)

K. Bulthuis, M.G. Carasso, J.P.J. Heemskerk, P.J. Kivits, W.J. Kleuters, and P. Zalin, "Ten Billion Bits on a Disk", IEEE Spectrum (August 1979).

K.M. Chandy, J.C. Browne, C.W. Dissly, and W.R. Uhrig, "Analytic Models for Rollback and Recovery", IEEE Transactions on Software Engineering, Vol. SE-1, No. 1 (March 1975).

K.M. Chandy, "A Survey of Analytic Models for Rollback and Recovery Strategies in Database Systems", Computer, Vol. 8, No. 5 (May 1975).

E.F. Codd, "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM, Vol. 13, No. 6 (June 1970).

E.F. Codd, "Entending the Data Base Relational Model to Capture More Meaning", Proceedings of the International Conference on Management of Data, ACM SIGMOD, Boston, Mass. (May 1979).

P.A.V. Hall, J. Owlett and S.J.P. Todd, "Relations and Entities", in Modelling in Data Base Management Systems, edited by G.M. Nijssen, North-Holland Pub. Co., Amsterdam, New York, Oxford (1976).

J.B. Harvill, A Programming Language Model for an Operand State Saving Computer, Ph.D. Dissertation, Southern Methodist University (Jan. 1978).

J.B. Harvill, "Functional Parallelism in an Operand State Saving Computer", Proceedings of the Fourth Workshop on Computer Architecture for Non-Numeric Processing, Blue Mountain Lake, New York, ACM and IEEE (August 1978).

A. Kaneko, Y. Nishihara, K. Tsuruoka, and M. Hattori, "Logical Clock Synchronization Method for Duplicated Database Control", Proceedings of the First International Conference on Distributed Computing Systems, Huntsville, Alabama, IEEE Computer Society (October 1979).

G.S. Kenney, D.Y.K. Lou, R. McFarlane, A.Y. Chan, J.S. Nadan, T.R. Kohler, J.G. Wagner, and F. Zernike, "An Optical Disk Replaces 25 Mag Tapes", IEEE Spectrum (February 1979).

W. Kent, Data and Reality, North-Holland Pub. Co., Amsterdam, New York, Oxford (1978).

L.J. Laub, "Optical Mass Storage Technology", Proceedings of the 1980 Workshop on Computer Architecture for Non-Numeric Processing, Pacific Grove, California, ACM and IEEE (March 1980).

J.M. Nielsen, Memory and Amnesia, San Lucas Press (1958).

D.P. Reed, "Implementing Atomic Actions on Decentralized Data" Proceedings of the Seventh Symposium on Operating Systems Principles, ACM SIGOPS, Pacific Grove, California (December 1979).

D.J. Rosenkrantz, "Dynamic Database Dumping", Proceedings of the International Conference on Management of Data, ACM SIGMOD, Austin, Texas (May 1978).

J.M. Smith and D.C.P. Smith, "Principles of Database Conceptual Design", Proceedings of the NYU Symposium on Database Design, New York, New York (May 1978).

R.H. Thomas, "A Solution to the Concurrency Control Problem for Multiple Copy Data Bases", Proceedings of COMPCON (spring 1978).

B.J. Underwood, "Attributes of Memory", Psychological Review, 76 (June 1969).