Title
A DATA MODEL AND THE DESIGN OF A SCIENTIFIC DATABASE MANAGEMENT SYSTEM

Permalink
https://escholarship.org/uc/item/8k5282fq

Author
Chan, P.S.

Publication Date
1985-02-01
A DATA MODEL AND THE DESIGN OF A SCIENTIFIC DATABASE MANAGEMENT SYSTEM

P.S. Chan

February 1985

TWO-WEEK LOAN COPY

This is a Library Circulating Copy which may be borrowed for two weeks.
DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.
A Data Model and the Design of a Scientific Database Management System

Paul S. Chan

Computer Science Research Department
University of California
Lawrence Berkeley Laboratory
Berkeley, California 94720

February 1985

This research was supported by the Applied Mathematics Sciences Research Program of the Office of Energy Research, U.S. Department of Energy under contract DE-AC03-76SF00098.
A Data Model and the Design of a Scientific Database Management System

Paul S. Chan
Lawrence Berkeley Laboratory
University of California
Berkeley, California

Abstract

A data model consists of two levels is discussed in this paper. The logical level integrates the n-dimensional concept with the graph concept into a single data structure. To satisfy the requirement of time variation of data definitions, the idea of effectivity is incorporated. At the physical level, the data model partitions the n-dimensional data space into subspaces which can be accessed randomly and processed in parallel.
Introduction

The term "Scientific Database" is somewhat misleading. When we are speaking of scientific data bases in this paper, we are referring to a class of data bases whose data structure and usages are quite different from those of traditional business applications. These data structures and usages are often found in, but not limited to, scientific applications. However, there are also data bases in scientific applications whose data structures and usages are similar to the traditional hierarchical, network and relational models.

According to a classification scheme given by the committee on Data for Science and Technology of the International Council of Scientific Union, there are three main categories of Scientific and Technical data [1]:

Class A - Property data. These are repeatable measurements on well defined systems. For example, physical and chemical data resulting from measurement of well understood properties of systems of known compositions.

Class B - Observational data. These are results of time or space dependent measurements that cannot be checked by remeasurement. For example, data from biology, geoscience, and environment monitoring, etc. all belong to this category. This class of data is often used for research in physical or biological science.

Class C - Statistical data. This class of data is also time or space dependent. For example, demographic data, energy consumption data, health statistics, etc. all belong to this category. This class of data is often used in social science research or economic planning.

Yet another class of data which is relevant to scientific data management are results or temporary results of mathematical modelling of physical systems. This kind of data is also space and time dependent. From the data management view point, it is not important whether the data can be checked by remeasurement. However, the data organization may be dependent on whether the data is time and space related. Several examples of scientific data bases and their characteristics are discussed in [14], [15].

Requirements of Scientific Data Management

1. The data in class A are often associated with bibliographic information such as author names, publication date and subject key words. A number of specialized systems such as FRAMIS (Lawrence Livermore Laboratory), RECON (Oakridge National Laboratory), SPIRES (Standford University), have been developed for this kind of data bases.

2. Both class B and C data are space and time dependent. The data measured or collected can be viewed as snap shots of the state of the system. As data are collected over a long period of time, the configuration of the system or the environment in which data are defined and collected may change with time. Therefore, the data base must contain not only the snap shots, but the corresponding data definition as well. In contrast, when data values of a conventional data base change with time, new data values replace old data values as the changes occur. As the data definition changes, the data base is restructured to reflect the new data organization. There is only one operational data definition in existence at any time. Even though this requirement is not unique to scientific data bases, we are dealing with this kind of data bases mostly in the laboratory, we'll focus on their requirements in the following discussions.
3. Most scientific data are n-dimensional oriented. The traditional hierarchical and network model are not compatible with the n-dimensional data structure. N-dimensional data are commonly stored as flat files. For example, a four dimensional data space is shown below.

\[ F(t_i, x_j, y_k, v_l) \quad \text{for} \]
\[ i = 1, 2, \ldots, I \]
\[ j = 1, 2, \ldots, J \]
\[ k = 1, 2, \ldots, K \]
\[ l = 1, 2, \ldots, L \]

where \(x, y\) and \(t\) designates the positions and time, \(v\) are measurements taken. There are two common approaches in organizing this data using traditional data management systems. One approach is to store the data in a file which consists of \(L + 3\) data elements. The data elements are \(t, x, y, v_1, v_2, \ldots, v_L\). (e.g. \(v_1\) = pressure, \(v_2\) = temperature, .. etc). The disadvantage of this approach is that it requires a large amount of storage space to store all possible combinations of data values of \(t_i, x_j, y_k\).

Another approach is to store the data in a file which consists of \(I\) records and \(J*K*L\) data elements. In other words, there is one data record for each point in time. This approach requires no space to store the data values of \(t, x, y\), but it may have a very large number data elements. Expressing queries may be very difficult. From the user's point of view (i.e. logically), the first approach is better because it is much easier to express queries. From the system's point of view (i.e. physically), the second approach is better, because it requires minimum storage. An ideal data model is one which has the benefit of both approaches but not the disadvantages.

4. After scientific data are collected, they may be analyzed many times with different access patterns. A set of scientific data may also be used by more than one program which have different access patterns. Different programs may use different coordinate systems. For example, one program may use cartesian coordinates and another may use spherical coordinates. The data model must provide flexible data organization so that average access time can be minimized.

5. The data processing rate requirement in scientific applications is often much higher than those of business applications. For example, weather prediction, nuclear reaction and climatology simulations often take days and weeks of processing. From the data management view point, there are two ways that higher processing rate can be achieved. One way is to provide very efficient access mechanism and the other way is to access and process data in parallel. Therefore, A data model which can take advantage of parallel processing capability is needed. The best algorithm used to solve a problem and the corresponding data organization may be machine architecture dependent, it is important to minimize the effect of this kind of dependence so that the same program can be used on a large variety of machines.

6. The amount of data collected in scientific experiments and mathematical modelling are constantly increasing. Hence the data storage capacity requirement is also much higher in the scientific environment. To facilitate high performance access in a high data volume environment, the data model must allow data to be organized in a multi-level storage hierarchy. The system should be capable of managing data in high speed processor memory as well as high volume sequential storage such as tape or mass storage systems. The system must also provide data independence so that changes in storage organization do not affect the program.
7. Most large scale experimental research programs require very expensive specialized facilities. The amount of data generated is often enormous. It is not practical to duplicate these facilities or all the data generated in more than one location. To achieve the objective of resource and data sharing, a distributed data base management system which operates in a network environment is needed. In scientific applications, data are usually generated and updated from one source. To share data in a remote location, subset of the data can be copied and processed locally. If the amount of data to be shared is large, it can be processed remotely and the results returned to the local site. Scientific data are not update intensive after they are collected. Even if updates are necessary, data consistency is desirable but not critical as in the business environment. These facts greatly simplifies the difficulties in managing a distributed scientific data base. On the other hand, data can only be shared if their existence are known to other users. Keyword search similar to those of class A scientific data would be very useful because data can be accessed and cross referenced (joined) without apriori knowledge of the data structure. The data model we are seeking is one which can facilitate query by keywords with minimum effort on the users behalf to catalog the data.

8. If the benefit of data sharing is to be realized, it is necessary to have as many users and their data integrated into one system as possible. This requirement together with the fact that all versions of the data definition must be saved imply that the data base definition may be quite substantial. The data definition may in fact be constantly increasing such that itself must be stored as a data base.

9. There are other requirements such as data security, integrity, etc. We have not fully understood these requirements so that they can be considered in our design. We feel that the treatment of these problems may also be quite different from those of the business applications. For example, in statistical data bases, data security may be defined in terms of levels of aggregation and combinations of data sets.
The Logical Data Model

A directory driven system SUBJECT [9] was developed at the Lawrence Berkeley Laboratory in 1980 for social economic applications. At that time, the motive was to provide a friendly environment for social and biological science research. Later projects based on the same approach are described in [16],[17]. Since then, we have been experimenting with many other scientific and statistical data bases to gain further insight into the problem of scientific data management. We have continuously refined the data model as new requirements are understood. In this paper, we'll try to summarize our finding and to give examples to illustrate the concept.

The logical model of scientific data is a graph $G(N,L)$ consists of a set of nodes $N=\{n_1,n_2,\ldots,n_i\}$ and a set of links $L=\{l_1,l_2,\ldots,l_k\}$ such that each link $l_k$ is identified with an ordered pair $(n_i,n_j)$ of nodes. The number of links incident out of a node $n_i$ is called the out degree and the number of links incident into $n_i$ is called the in degree. A positive integer $D_i (D_i \geq 0)$, which designates the dimension of each node, must be associated with each node. One or more effectivity range can be associated with a link $l_k$. Each effectivity range consists of a pair of numbers $(x_k,y_k)$ such that $x_k \leq y_k$. The effective range may be either inclusive or exclusive, designated as $+(x_k,y_k)$ or $-(-x_k,y_k)$. If no effectivity range is associated with a link, it is assumed to be $+(-\infty,\infty)$. In other words, it is always effective. A graph representing the logical structure of a scientific data base has the following properties:

1. If $D_i=0$, the out degree of $n_i$ must be 0. In this case, the node $n_i$ is called a terminal.

2. If $D_i=1$, the out degree of $n_i$ must be greater than 1.

3. If $D_i>1$, the out degree of $n_i$ must equal to $D_i$. Moreover, if effectivity ranges are associated with one of the out links of $n_i$, then all other out links of $n_i$ also have the same effectivity ranges.

4. If more than one effectivity range are associated with two nodes $n_i$ and $n_j$, then the effectivity ranges of the link between $n_i$ and $n_j$ must not overlap.

5. If $n_j$ is not a terminal, then all the links of the sub-graph originated from $n_j$ also have the same effectivities as the link between $n_i$ and $n_j$. If different effectivity ranges are associated with the links in the sub-graph, then the effectivity ranges in the sub-graph must be within one of the effectivity ranges associated with the link between $n_i$ and $n_j$.

6. The in link of a node $n_i$ must not be the out link of a node $n_j$ which is directly or indirectly connected to $n_i$ (i.e. There must be no cycles or self feeding loop).

Basically, this model integrates the n-dimensional concept with the graph concept into a single data structure. This data structure allows n-dimensional data structures within hierarchies (graphs) and hierarchies within n-dimensional structures. This recursion can continue for as many levels as needed. As shown in the examples below, hierarchies or graphs are necessary in order to represent complicated relationships of the environment or configuration from which data is generated. The concept of effectivity is incorporated into the model so that the logical structure can change with time or other conditions. Notice that effectivity ranges are associated with links (rather than nodes). This is because a node can be shared by several links and each link may have a different effectivity range.

Nodes are identified by names. Node names can be expressed in English sentences consisting of as
many words as needed. Nodes can also be referenced by keywords. Keywords are selected (e.g. marking them with quotes) words in the node names. The use of keywords allows node names to be searched and referenced without knowing the node content and organization. Keywords also allow data to be cross referenced. The physical characteristic of data such as the size and type of data, the storage device and location, are specified in the physical model.

The following example illustrates how time variation of data is represented in this model. An experiment using two clusters of detectors A and B, was set up in January 1980 to measure signals generated every second, 24 hours a day. Each detector has 6 channels, each measuring a different signal. On the 15th of January, the system was reconfigured to have to a new cluster of detectors C while the cluster B would no longer be used. No data was taken on that date. The system resumed operation on the 16th till the 31st. The logical structure of data is represented as follows.
The dimensionality of each node is shown in a square bracket except for terminals. The dimensionality of terminal nodes is not shown, it is defaulted to be 0. Two effectivity ranges are specified in this example. The effectivity range (1,14) and (16,31) indicate the operational date range of detector cluster B and C respectively. The structure as shown represents two 3 dimensional data spaces. One is 31 days by 6 channels by 4 detectors (because only detector cluster A and B are effective) and the other one is 31 days by 6 channels by 5 detectors (because only detector cluster A and C are effective).

To be precise, the effectivity ranges (1,14) and (16,31) should also be assigned to the links from day 1 to 14 and from day 16 to 31 respectively. The effectivity -(15,15) should be assigned to the link day 15 to indicate that no data was collected on the 15th. But it makes the structure much more complicated. The structure as shown would be sufficient as long as the physical data can be mapped to the exact position of the logical data space. This graph can be changed and new effectivity ranges assigned as needed. In scientific applications, effectivity ranges are usually related to date or time. But they can also be given other significance as the condition of variations.

The Data Language of the Logical Model

The following example illustrates how different sets of scientific data are integrated into a single data base.

---

The diagram illustrates the integration of different sets of scientific data into a single database. The data sets include:
- Energy Consumption
- Cancer Death
- Air Pollution
- Fuel Type
- Time
- State/Country
- Sex by Age by Race
- Air Pollutants
- Nuclear Coal
- Year
- AL CA NY
- Sex Age Race
- CO NO
- MF 12.99 A B W

The diagram shows how these data sets are structured and interconnected, with each node representing a dimension and the links indicating the operational ranges.
The example shown has three sets of scientific data, Energy consumption (3-D), Cancer death (4-D) and Air pollution (3-D). These three sets of data share two common dimensions, Time and State/County. The sub-structure of shared nodes is shown only once in the figure. Internally, the nodes are shared.

The most interesting feature of this data model is that an interactive retrieval language is part of the data model itself. A graph with the above mentioned properties defines the logical data space of the data base. A mechanism is provided for the users to scan all keywords and data names (i.e. node names) in the structure. After nodes of interest are identified, the structure can be entered at any point by specifying the name of the node and then transversed up and down for data selection or aggregation. The followings clarifies the semantics of data selection.

1. When a node is selected, a data sub-space is specified. If no other node is selected, the data sub-space specified is the target of retrieval. In the example given, selecting the node nuclear is the same as:

   Select fuel type=nuclear for all the states and all the years in energy consumption.

2. When more than one node are selected, the data sub-space specified may overlap. If there is no overlap, then the union of the two data sub-spaces is the target of retrieval. For example, if both nuclear and coal are selected, it is equivalent to the query:

   Select fuel type=nuclear or coal for all the state and all the years in energy consumption.

3. If the data sub-spaces selected overlaps, then the intersection of the data sub-spaces is the target of retrieval. For example, if the nodes nuclear, coal, 1980, 1981, Jan, Feb, CA, NV are selected, it is equivalent to the query:

   Select (fuel type=nuclear or coal) and ((year=1980 or 1981) and (month=Jan or Feb)) and (state=CA or NV) in energy consumption.

4. If more than one set of scientific data are selected and if nodes shared by these sets are selected, it is equivalent to a join condition. For example, if nuclear, coal, energy consumption, CO, NO, air pollution, and the shared nodes CA, NV and 1980, 1981 are selected, the query is the same as:

   Join state and time in energy consumption to state and time in air pollution.

   Select (fuel type=nuclear or coal) and (year=1980 or 1981) and (state=CA or NV) and (air pollutants=NO or CO).

The advantage of this method of query is that no syntax or data names have to be remembered. However, there are queries which cannot be expressed easily. Moreover, this method of query cannot be used in computer programs. Therefore, a syntax oriented query language is also needed.

To access data in the data base, two levels of mapping are needed. The first level maps the nodes selected to the positions of the co-ordinates of each dimension in the n-dimensional logical data space. The second level maps the logical data sub-space defined by the positions of these co-ordinates into the storage locations of the devices containing the data. If the sub-structure selected contains more than one effectivity range, the position of the coordinates of the logical data space associated with each effectivity range will be determined independent of each other.
The physical data associated with each effectivity range can then be accessed.

*Equivalent Structures and Parallel Processing*

There are more than one way that a set of physical data can be represented logically. Two logical structures are equivalent if they contain the same data space. For example, the following structures are equivalent.

![Diagram of Age by Race by Sex](image)

Figure A.

![Diagram of Race by Sex](image)

Figure B.
From the access efficiency viewpoint, the structure in Figure A is better because the positions of the co-ordinates can be calculated. Whereas in Figure B, the position is determined by enumerating all possible paths from age to sex.

It is not uncommon that some scientific data have very complicated dimensions. For example, the geographic dimension may consist of state county and tract levels. There are a total of about 4000 counties in the US. On the average, each county has about 1111 tracts. In such cases, to search down a structure to determine the position of a node (coordinate) may be quite time consuming. The position of a node in one dimension is independent of the position of a node in another dimension. Therefore, if the structure of each dimension is stored on storage devices which can be searched in parallel, then the position of the nodes in each dimension can be determined independently. Furthermore, if the coordinates in a dimension can be structured hierarchically as in the case of state/county/tract, substructures can also be stored in parallel storage devices. Searching of a node can then be carried out in parallel and the results summed to determine the position of the coordinate.

N-dimensional table look up is very common in mathematical modelling of physical systems. For example, the "equation of state" \(^{10}\) is needed in hydrodynamics modelling as well as other physical processes. Functions describing the equation of state can be stored as N-dimensional tables. The use of such tables can be seen from the following example. Given the temperature and density of a material, related physical quantities such as pressure, energy, etc. can be estimated by interpolation of the values in the table. Very often, the pressure and energy for a range of temperature and density (say, 1000-20,000 points in each case) are needed simultaneously. The table look up and interpolation constitute a very substantial portion of the processing time. This kind of modelling can easily take ten or more hours per production run depending on the speed of the machine and the degree of accuracy required. Obviously, parallel processing can greatly contribute to the improvement in processing time. There are many ways that an n-dimensional table can be organized logically and physically. Two alternatives are given as follows.

![Figure C](image1)

![Figure D](image2)
the conceptual organization of A, B and C are simply N x N matrices. At the logical level, it is organized as equivalent structures which reflect the data structure requirements of the machine and data operations. The mapping between the virtual machine of unlimited processors and the physical machine is done at this level. In this example, the algorithm of each class of machine is different, hence a different mapping is needed for each class. The physical level specifies how data are actually stored in storage device or processor memory. If a complete set of high level data manipulation operations (such as matrix multiplication in this example) can be defined, then scientific applications can be programmed independent of machine architecture.

The Physical Data Model

One of the most computation and data intensive mathematical modelling problems is the simulation of three dimensional fluid flow [13]. The geometry is defined by a grid system where each node point is assigned an x, y, z location. The grid is almost always non-uniform because body conforming mesh lines and grid clustering are used for better resolution. This domain must be mapped into a computational grid space which is uniform and of unit length so that equispaced finite difference formulas can be used. The solution consists of solving a large number of simultaneous equations for each time step. The algorithm requires a very large temporary storage capacity, even if the number of grid points is only moderate. To achieve reasonable accuracy, the temporary storage requirement can easily exceed processor memory capacity. The bottleneck of this problem is in disk to memory data transfer. Even if the computer has a large number of processors, they cannot be operated in parallel unless the required data can be fed to them continuously. It is crucial that data be mapped onto disk such that the largest amount of data be interchanged in the course of one revolution. It is also crucial to minimize these interchanges and the access conflicts between processors. These kind of problems are to be handled by the physical level of the database management system.

As mentioned above, two levels of mapping are necessary. The first level maps the nodes selected into the positions of the coordinates in the n-dimensional data space represented by the logical structure of each effectivity range. The second level maps the positions of these co-ordinates into the locations of the storage devices/processor memory in which the data are stored. The most straight forward technique for the second level of mapping is known as array linearization. However, more sophisticated techniques are necessary for the following reasons:

1. In computer programs, if all elements of an n-dimensional array can fit in main memory, the access time of any access pattern are about the same. This is not true if data are stored on disk or other secondary storage devices. Data stored on secondary storage devices according to the order of array linearization may be efficient in one access pattern but very poor in others. Hence, programs which take advantage of virtual memory capability may suffer severe performance degradation.

2. The data space may not be full. In other words, data values may not exist for all possible combinations of coordinates from each dimension. If data are stored according the array linearization technique, storage space cannot be fully utilized.

3. To support a multi-level of storage hierarchy, it is important to store data which are used frequently in high speed storage. Cache memory have widely been used in high speed CPU based on two assumptions. (1) Memory addresses that have been referenced recently are likely to be addressed again in the near future. (2) The locality of memory references are likely to be close to what is being referenced. We believe these two assumptions are basically correct in scientific data analysis. But unlike computer programs which are composed of linear sequences of instructions, the locality of reference in an n-dimensional data space may not be linear.
4. The array linearization technique does not provide enough flexibility to organize data for parallel processing.

Some work have been done in treating a data file of n-attributes as an n-dimensional record space. The n-dimensional record space can be partitioned into n-dimensional sub-space called cells [2], [3]. The following summarizes the n-dimensional file concept.

Each attribute of the file corresponds to a dimension of the n-dimensional record space. Initially, the coordinates of each dimension are chosen such that the n-dimensional record space is partitioned into cells with equal number of records. To make this partition possible, the records must be pre-sorted n ways. The coordinates which define the cell partition of each dimension are stored in tables such that given any combinations of coordinates selected from each dimension, the cells which contain the records can be identified.

One or more cells can be assigned to physical storage units call buckets. In order to process range queries efficiently, it is required that cells assigned to each bucket form the shape of an n-dimensional rectangle. Since the size of a cell must be less than or equal to the size of a bucket, the bucket size directly affects the cell directory size. As new records are added to the file, cell expansion may cause bucket overflow which trigger the refinement of the partition. The coordinates which define the partition and the cell directory must then be updated accordingly.

The cell directory can be viewed as a file of n attributes. There are basically two approaches in organizing the cell directory, the tree organization and address calculation. Much work have been done on organizing a multi-key file as a tree structure. It includes the QUAD tree [4], the K-D tree [5], Quintary tree [6], multi-dimensional B-tree [7], etc. The address calculation approach include the GRID file [3] and other multi-key hashing techniques [8].

The address calculation approach is more efficient than the tree organization approach. But the address calculation technique requires a table to maintain the values of the coordinates which define the partition for each dimension. If the number of distinct key values is small, the table can be stored in memory. But if it is too big to be stored in memory, then the tree structure may be a better choice. Another advantage of the address calculation technique is that it is symmetric. In other words, there is no distinction between primary and secondary keys. The access efficiency of keys in a tree structure is dependent on how the keys are organized.

In applying the n-dimensional file concept to our data model, we have made the following design decisions.

1. The n-dimensional file concept is applicable to the physical level of the data model only. An interface between the logical and physical level must be provided.

2. The n-dimensional data space of a set of scientific data may be dense or full. Storing each number in the n-dimensional data space as a record which carries a data value for each dimension may be very uneconomical. The other alternative is to position the number in the cell so that the value of each dimension can be derived. This method also cause some waste of storage space unless the data space is full. To achieve full storage utilization, cells can be compressed for storage and decompressed for retrieval. This method allows compressed data to be randomly accessed without decompressing the entire file.

3. A disadvantage of the n-dimensional file concept is that storage space cannot be 100% utilized. If data are added to the data base in a random sequence, cell overflow may cause a storage bucket to split from a full bucket to two half filled buckets. Simulation result shows that a 70% load factor can be achieved. Scientific data are likely to be created in predictable sequence, hence buckets
can be filled sequentially. Therefore it is possible to achieve close to full storage utilization.

4. To satisfy the requirement of high processing rate, the data model must take advantage of parallel processing capability and multi-level storage hierarchy. To minimize access contention, cells to be processed in parallel should reside on separate devices/memory modules. On the other hand, cells to be processed sequentially should reside on the same device/memory module and be located as close to each other as possible. A flexible method for assigning cells to buckets is needed.

5. To satisfy high data volume requirement, the data model must take advantage of a multi-level storage hierarchy. As mentioned before, it is assumed that the loci of reference is likely to be near neighbors. In an n-dimensional data space, near neighbors may be in any one of the 2n directions. Moreover, a different access pattern may access a different set of near neighbors. Therefore, a flexible method for specifying the size and dimension of the cell is needed in order to minimize average access time.

6. It is also assumed that some cells may be used more often than others and the usage pattern may change with time. The cell directory and the cells used frequently must reside on high speed storage devices. After the selection criteria at the logical level are mapped into the positions of the coordinates of each dimension, the cell directory is then consulted to determine the buckets in which the cells reside. The buckets are then searched sequentially to locate the cells and the cells are decompressed to locate the data values. The size of the directory is inversely proportional to the total number of cells. A large cell directory with small buckets allows small units of data to be randomly accessible. If the retrieval pattern requires large volume of sequential access, then a small cell directory with a large bucket size may suffice.

7. Business data are likely to be created or deleted in a random sequence. To satisfy retrieval requests which may occur immediately after data are added or updated, the data must be organized for efficient access on-line. As mentioned above, scientific data are likely to be generated in a predictable sequence. Immediate availability after data are created is not as important, but the same data may be analyzed many times. Therefore, data can be stored in the same sequence as they are generated and later re-organized for efficient retrieval.

**The Physical Data Definition**

There are basically four variables which can be adjusted to satisfy the processing speed and storage space requirement of an application. These four variables are the size and dimension of the cell, the bucket size and the cell to bucket assignment.

The cell size can range from storing all numbers in an n-dimensional data space to one number per cell. The cell dimension can range from n to 0. In the first extreme, to access a number, the entire file must be processed sequentially. In the other extreme, a random access is needed every time a number is retrieved. These two extremes are less than optimal in most situations. The ideal organization is one which allow random access to a bucket containing all data needed. To maximize access efficiency, buckets should constitute storage units which can be accessed sequentially. The bucket size determines the amount of data that can be processed sequentially after the bucket is located. The cell to bucket assignment, and the size and dimension of the cell determines how data will be processed sequentially. N-dimensional data spaces are commonly stored as two dimensional slices (cells). This organization would be very inefficient if an application requires a number from each slice. Given these four variables, an optimal mix of sequential and random access can be achieved by choosing the right parameters. As mentioned, the total number of cells and hence the size of the cell directory is inversely proportional to the cell size. Therefore, these variables also provide a time space trade off. To completely define the physical data model, the
followings must be specified.

1. The logical and physical data space association.

Any non-terminal node in a graph can be assigned to a physical data space. To support multiple logical organizations, more than one non-terminal node in a graph can be assigned to a physical data space. This association is maintained in a table $T$, which can be expressed as $P = T(L_k)$, where $P$ is the physical data space ID, $L$ is the name of a non-terminal node in the graph structure, $k$ is the effectivity range of the sub-structure originated from $L$. There is one physical data space associated with each $k$ in $L$. It is assumed that the type and size of all data values in the physical data space are the same. This information is also contained in the table.

2. Association of the co-ordinates of each dimension between logical and physical data space.

To simplify the mapping between the logical to physical data space, the physical data space is always expressed in cartesian coordinate. If the coordinates of each dimension of the physical and logical data space coincide, no mapping function is needed. But if more than one effectivity range is associated with the logical structure, or if the coordinates of the logical structure are expressed in another coordinate system such as cylindrical or spherical, or if the dimensionality of the logical and physical data space are not the same, then a set of mapping functions or tables is needed to translate the coordinates of the dimensions of each logical data space to that of the physical data space. These mapping functions can be expressed as follows. Given an effectivity range $k$, the logical space is defined as $d_1, d_2, \ldots, d_m$. Each dimension $d_i$ has a maximum number of coordinates $D_i$. Similarly, the physical data space is defined as $e_1, e_2, \ldots, e_n$. Each dimension $e_i$ has a maximum number of coordinates $E_i$. For each physical dimension $e_i$, a function $f_{i_k}$ is defined over the dimensions of the logical space $d_1, d_2, \ldots, d_m$ as:

$$e_i = f_{i_k}(d_1, d_2, \ldots, d_m)$$

$$i_k = 1, 2, 3, \ldots, n_k$$
3. The cell partition.

There are more than one way that cell partitioning can be specified.

a. If the data space is dense or full, then cell partitioning can be specified as a set of integers $C_i$, such that

$$1 \leq C_i \leq E_i, \quad i = 1, 2, \ldots, n$$

$E_i$ is the range of the $i$th dimension, which can be determined if $f_{i}$ is defined.

If $C_i = E_i$, then the $i$th dimension is not partitioned.

If $C_i = 1$, then the dimension of the cell will be reduced by 1.

A commonly used method, which stores two dimensional slices of the n-dimensional data space sequentially is expressed as,

$$C_{1i} = E_{1i}, \quad C_{2i} = E_{2i}, \quad C_{3i} = C_{4i} = \ldots = C_{ni} = 1.$$ 

b. If the data space is sparse, cell partitioning can be specified as an integer $B_k$, where

$$1 \leq B_k \leq E_{1i} \cdot E_{2i} \cdot \ldots \cdot E_{ni},$$

is the number of data values per cell.

If $B_k = E_{1i} \cdot E_{2i} \cdot \ldots \cdot E_{ni}$, then the data space is not partitioned.

If $B_k = 1$, then the entire data space is in one cell. In this case, a very large bucket (sequential file) such as a tape is needed.

Given $B_k$, the coordinates which partition the physical data space can be determined. These coordinates are likely to be irregularly spaced. In other words, they cannot be expressed as fixed increments and must be stored in tables or index.

c. A combination of these two methods of specification can be used. A selected number of $C_i$ can be specified and then the remaining partitions can be determined given $B_k$. This method of cell partition is both easy and flexible.

4. Bucket to device assignment.

In order to maximize access efficiency, each bucket must constitute contiguous storage units of the storage device. For example, if the storage device is a disk, then the bucket size may be chosen as multiples of sectors, tracks or cylinders. To support a multi-level storage hierarchy, the bucket size should be device dependent. Buckets are identified by bucket ID which consist of a device ID and a bucket number. Bucket numbers are assigned to buckets sequentially according to the physical location of the bucket. The device ID also identifies the device type and the bucket size.
5. The cell to bucket assignment and the cell directory.

The association of cells to buckets is specified by the cell directory. The algorithm which assigns cells to buckets is based on the access pattern and the machine/storage device organization. If the access pattern is not known, a reasonable assumption is that cells which are close to each other are likely to be accessed together. By treating each cell as a point and by linearizing the n-dimensional cell space, a cell number can be assigned to each cell. The cell directory may be directly accessible by cell number or a B-tree. Each entry in the cell directory consists of a cell number and the bucket ID in which the cell resides. If there are very few empty cells, then a direct addressable cell directory is most efficient. Otherwise, a B-tree cell directory is more economical because empty cells can be omitted from the directory. A very large data space may reside on multiple levels of devices with the most frequently accessed data in highest speed storage.

Summary

Significant progress has been made in data management of business applications in the last decade. Unfortunately, the requirements of scientific data management are not exactly the same as those of business applications and the results can not be directly applied. Time variation of data definition and the multi-dimensional data structure of scientific data require data to be defined in two levels, logical and physical. The logical model integrates the n-dimensional concept with the hierarchical (graph) concept into a single structure. The logical structure can vary with time by means of controlling effectivity range. The physical model partitions the n-dimensional data space into data sub-spaces which can be accessed randomly. To achieve high performance in a high data volume scientific environment, this data model is designed to take advantage of parallel processing and multi-level storage hierarchy capability. To maximize sharing and utilization of scientific data without apriori knowledge of the data base organization, the idea of query by keywords is proposed. Concurrency control issues are not very important because scientific data are relatively static and the objective of data sharing is not quite the same as in the business environment. The algorithm and data organization of a program may be machine architecture dependent. To achieve data independence, three levels of data definition are suggested.

There are still many open problems such as the definition of data manipulation operators upon which new operations can be built, alternative rules for assigning cells to buckets, optimization between random and sequential access, optimal cell size/dimension and time space trade off, sophisticated scientific data manipulation operations and language, data security in the scientific environment, etc.

It is hoped that the availability of an advanced technology in scientific data management will help create new breakthroughs in scientific research.

Acknowledgements

The author is indebted to his colleague Arie Shoshani for his comments and extensive editorial assistance.
References

[1] Rumble J. and Hampel V.

Database management in science and technology.
North-Holland, 1984

[2] Liou J.H. and Yao S.B.

Multi-dimensional clustering for database organization.
Inf. Syst. 2(1977)


The grid file: An adaptable symmetric multi-key file structure.
In Trends in Information Processing System.
Proc. 3rd ECI Conference. (1981)


QUAD Tree - A data structure for retrieval on composite keys.
Acta Inf. 4(1974)


Multi-dimensional binary search tree in database applications.
IEEE Trans. in Software Engineering
SE-5, 4(July 1979)


Quintary Trees:
A file structure for multi-dimensional database systems.


The K-D-B Tree.
A search structure for large multi-dimensional dynamic indexes.
In Proc. SIGMOD Conference 1981. ACM

[8] Otto, E.J.

A Mapping function for the directory of a multi-dimensional extendible hashing.
Proc. 10th International Conference on Very Large Data Bases 1984
[9] Chan, P. and Shoshani A.

Subject:
A directory driven system for organizing and accessing large statistical databases.
Proc. 7th International Conference on Very Large Data Bases. 1981

[10] Dubois, P.

Swimming upstream:
Calculating table lookups and piecewise functions.


[12] Ostlund N.S., Hibbard P.G. and Whiteside R.A.

A case study in the application of a tightly coupled multiprocessor to scientific computation.

[13] Lomax H.

A fully implicit, factored code for computing three dimensional flows on the ILLIAC IV.


Characteristics of Scientific Databases.
Proc. 10th International Conference on Very Large Data Bases. 1984

[15] Bell, J.

Data Structures for Scientific Simulation Programs.
Proc. 2nd International Workshop on Statistical Database Management. 1983


Proc. 2nd International Workshop on Statistical Database Management. 1983

[17] Rafanelli M. and Ricci F.

Proposal of a Logical Model for Statistical Database.
Proc. 2nd International Workshop on Statistical Database Management. 1983
This report was done with support from the Department of Energy. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of The Regents of the University of California, the Lawrence Berkeley Laboratory or the Department of Energy.

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.