CHARACTERISTICS OF SCIENTIFIC DATABASES

Arie Shoshani, Frank Olken, and Harry K.T. Wong

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

ABSTRACT

The purpose of this paper is to examine the kinds of data and usage of scientific databases and to identify common characteristics among the different disciplines. Most scientific databases do not use general purpose database management systems (DBMSs). The main reason is that they have data structures and usage patterns that cannot be easily accommodated by existing DBMSs. It is the purpose of this paper to identify the special database management needs of scientific databases, and to point out directions for further research specifically oriented to these needs.

We discuss the different types of scientific databases, and list the properties identified for them. Examples applications are then analyzed with respect to the types of data and their characteristics, and summarized in two tables. Conclusions are drawn as to the preferable data management methods needed in support of scientific databases.

1. INTRODUCTION

This document is a result of numerous interviews with scientists, mostly from Lawrence Berkeley Laboratory, spanning several different scientific disciplines. The purpose of these interviews was to examine the kinds of data and usage of scientific databases in order to identify common characteristics among the different disciplines.

In the past, we have studied "statistical databases", which are databases that are primarily collected for statistical analysis purposes. A summary of work in statistical databases can be found in [Shoshani 82]. We expected that some of the observations and techniques developed for statistical databases will be useful for scientific databases. Indeed, we found this to be the case. The similarities are pointed out throughout this document where appropriate. It should not be surprising that common characteristics exist, because many scientific databases are often subject to statistical analysis. However, as discussed below, scientific databases have additional stages of data collection and analysis that introduce more complexity and challenges.

In section 2, different types of scientific databases are described. In order to describe the common properties between several example applications, a list of characteristics for the different types of data are described in section 3. In section 4, a representative example of a scientific application is delineated with respect to the list of characteristics. Additional examples have been similarly analyzed, but because of space limitations are not described in this paper. The description and analysis of these example applications were described in [Shoshani et al 84]. The characteristics of these example applications are summarized in two tables, which appear at the end of this document. In section 5 we discuss the implications of our observations to desirable database techniques for scientific databases, and propose areas for further investigation. Section 6 is a short summary section.

2. TYPES OF SCIENTIFIC DATA

The scientific databases described to us during the interviews were analyzed in order to identify similar data structures, data characteristics and data usage among different applications. We found it convenient to distinguish between different types of scientific data. The important features for each type were identified, and different examples of scientific data were categorized accordingly. In this section we describe the data types and their main features.

Proceedings of the Tenth International Conference on Very Large Data Bases.

Singapore, August, 1984

2.1. EXPERIMENT DATA

Most scientific data result from experiments and simulations. Data from experiments are usually measurements of some physical phenomena, such as the collision of particle beams, or the spectra generated by molecules in a strong magnetic field. Data from simulations typically result from complex computations derived by using values from the previous time interval. Both experiment and simulation data have similar characteristics, and therefore are considered jointly. In order to simplify the terminology used here, we refer to such data as "experiment data", regardless of whether they are experiment or simulation data. Experiment data can be classified according to three characteristics: regularity, density, and time variation.

Regularity refers to the pattern of the points or coordinates for which values are measured or computed. For example, in physics experiments, detectors are placed in a specific configuration. If the configuration describes a regular grid or some other geometric structure, the experiment is said to have (spatial) regularity. Similarly, many simulations assume some regular grid for which values are computed, and therefore have spatial regularity. In addition, if values are measured or computed at regular time intervals, then time can be considered as another regular coordinate of the data.

In general, regularity implies that a mapping between the coordinates of measured values and the storage locations of these values can be made by means of a computation (such as "array linearization", which is simply a mapping from multi-dimensional space to linear space, similar to FORTRAN array mapping). Therefore, in such cases it is not necessary to store the coordinate values with each measured data value, resulting in storage savings and fast random access. On the other hand, when spatial irregularity exists it is necessary to enumerate the data points, and store their identifiers with the data values.

Density indicates whether all the potential data points have actual values associated with them. For example, simulation data of fluid motion computed on a regular grid would have data values (for velocity, direction, etc.) computed for each point of the grid, and therefore the data is considered dense. On the other hand, in many experiments a large number of measurements that are below a certain threshold are discarded and never recorded. In fact, the level of sparseness can be quite high, i.e. only a small fraction of the potential data points have recorded values. For example, in physics experiments of colliding particle beams, the measured data is only for resulting sub-particles, which occur over a small portion of the detectors that are distributed in space.

Sparsity implies a large number of null values which may be compressed out. The compression technique chosen should depend on the access patterns to the data, such as whether the data are accessed sequentially or randomly. Access patterns are discussed in the next section.

Time variation refers to the change of coordinates over time; i.e. the points for which data values are measured or computed change their position from one time unit to another. For example, consider some material that is bent in the course of an experiment. Before the experiment starts a set of points is selected for measuring the material's behavior (such as stress, voltage, temperature). During the experiment the selected points may change their position as a result of the bending action. Time variation is a characteristic found mostly in simulations where a mesh of points are allowed to change their position over time during the simulation process. These simulation methods are generally called adaptive mesh techniques.

Time variation adds an important requirement. In addition to storing the coordinates of points for every time interval, it is necessary to maintain the relationships between the points as they existed in the original mesh. This is needed in order to be able to reconstruct the time sequence of points that correspond to the same original point, and in order to find neighboring points to a given point at any given time.

2.2. ASSOCIATED DATA

In addition to the experiment data discussed above, there exist data in support of the experiments, and data that are generated from the experiment data. Support data fall into two types which we call configuration data and instrumentation data. Similarly, generated data fall into three types: analyzed data, summary data, and property data. These types are discussed below. To distinguish these additional data types from the experiment data, we refer to them collectively as "associated data".

2.2.1. Configuration data

Configuration data are data that describe the initial structure of an experiment or simulation. For example, in simulating heat transfer through buildings, the building layout has to be described. Similarly, the configuration of an experiment describes the position of different devices and detectors. The configuration layout actually determines the regularity (or irregularity) of the experiment data mentioned above. Usually, it does not change in the course of the experiment or simulation. However, it can change between experiments or simulations. It is important to keep track of these changes and to associate the correct

configuration data with the corresponding experiment data.

2.2.2. Instrumentation data

Instrumentation data consists of descriptions of the different instruments and substances used in an experiment, and their changes over time. This data is crucial for the correct analysis of the experiment data. It includes information such as the pressure and temperature of a gas used in an experiment and their changes over time, drift of voltage over time, and the characteristics of detectors and devices as measured before each experiment or a series of experiments. It also includes the log of experiment operations, such as the time that a defective analog-to-digital converter was replaced, and who was in charge of it. Unfortunately, some of this information is collected into unrelated files and log books, thus making their association with the experiment data a tedious task that is prone to errors.

2.2.3. Analyzed data

The previous two data types are essential in order to support the analysis of experiment data. The analysis process produces many databases that also need to be managed along with their relationships to the experiment data they were derived from and to each other. The analysis process may require several steps. For example, in physics experiments of colliding particle beams, a preliminary histogram over the experiment data can be done in order to estimate parameters that are later used to interpret the calibration data of detectors in the next step of the analysis. For each collision, called an event, the tracks of sub-particles produced are reconstructed and kept in a database. From the track data, another database for the event data can be derived, describing the kind of sub-particles produced and their characteristics. Additional steps use databases from this and earlier stages to generate yet more data. It is important to capture the analysis process, the input and output databases of each step, and the relationships between the steps.

2.2.4. Summary data

Similar to "statistical" databases, which deal with statistical summaries (aggregations) of data sets, scientific databases are often aggregated. For example, in experiments of heat transfer in buildings, the amount of heat lost or gained can be averaged over several points of a wall, summed over entire rooms, or aggregated over days into months. Another example, is the generation of histograms from many experiments to determine the likelihood of a certain phenomenon. As in the case of statistical databases, there is a need to organize, search and browse collections of summary data, and to preserve their relationship to lower level data from which they were derived.

2.2.5. Property data

In any scientific field, the summary of information learned over the years is useful to the community at large. There is a substantial amount of work devoted to the organization and classification of properties of materials, substances, and particles. For example, there are several systems devoted to the storage and retrieval of chemical substance properties. Many property databases cannot now be accessed on-line. The data is only available in periodically published books, and may not be up-to-date. Property data is non-uniform: it contains numeric, text, and bibliographic data, as well as images and graphs. This is one of the reasons that for each scientific area special purpose systems have been developed. Data management systems that can deal with such diversity of data types are not generally available. In addition, because of the complex terminology involved with such data, sophisticated search and browsing capabilities are needed.

3. CHARACTERISTICS IDENTIFIED FOR SCIENTIFIC DATA

Using the classifications of data types described in the previous section, it was easier to identify common characteristics and usage of the data. For each classification we have looked for certain characteristics that seem to exist across scientific applications. These characteristics are described in this section. Since the characteristics of experiment data are not necessarily the same as those of associated data, they are described separately. In the next section, we describe example applications in terms of these characteristics. The terms that are used for each characteristic are shown in *italics* in the text below. The reader may refer to the leftmost columns of table 1 and table 2 for the list of characteristics of experiment data and associated data, respectively.

3.1. CHARACTERISTICS OF EXPERIMENT DATA

1) Identifier

The identifier is that part of the data that identifies each data point uniquely (also called a key). In the case of experiment data the identifier is usually a composite key of spatial coordinates and a time coordinate. Since the identifier has multiple dimensions, the characteristics of regularity, sparsity, and time variation, discussed in the previous section, apply naturally. The concept of a multi-dimensional identifier is similar to that of category attributes in statistical databases [Shoshani 82]. This concept is quite dominant in experiment data (as was the case with statistical databases) because the

data is mostly accessed with respect to its identifier. We expand on this point below in the section on access patterns.

The identifier is said to be *regular* if each of its dimensions are ordered in regular intervals. It is *sparse* if only a fraction of the points in the cross product of the dimensions have data associated with them; otherwise it is *dense*. *Time variation* implies that the coordinates of the identifier, regardless whether it is regular or irregular, vary over time.

2) Access pattern

Access pattern refers to the most typical forms of data access. For example, an analysis program may follow a track of a sub-particle, or a simulation program may need its nearest neighbors in order to calculate the next data point. Note that in these examples the access of points is relative to the (spatial) identifier coordinates, and not the measured or calculated data values. This is typical of the access pattern of experiment data. The reason for distinguishing between the different types of access patterns is that they imply different requirements for physical data base organizations, as discussed below in the implications section.

We distinguish between two aspects of the access pattern. The *access type* is the type of access of a single query (or a step of the computation). The *access sequence* refers to the relationship between queries, i.e. whether the selection of a query depends on previous queries.

2a) Access type

There are three access types that we found useful to identify. An *exact match* means that the identifier of a point was specified precisely in the query. A *range* type implies that a range of possible points were identified. Since the identifier is multi-dimensional, each dimension is involved in the specification of the range. A *proximity* type indicates that the neighboring points around a given point are desired.

2b) Access sequence

Given a query of a particular type, the access sequence indicates whether the identifier(s) of the next query relate to the identifier(s) of the previous query. A *local* access sequence implies that the identifier(s) of the current query are close to the identifier(s) of the previous query. For example, following a particle track involves a local access sequence, since each successive point is close to the previous point. A *non-local* access sequence means that there is no relationship between the identifiers of successive queries. In a *linear* access sequence, the sequence of the identifiers of successive queries follows successive intervals of the dimensions of the identifier. For example, following the points of a mesh according to the regular intervals of the dimensions of the mesh is considered a linear access. An *arbitrary* access sequence indicates that the order of processing the data points is unimportant. Such access is usually used when the entire data set (or some large subset) need to be processed for analysis or summary statistics.

Access sequence should be thought of in conjunction with access type. For example, searching for a particular point in space where this point is not related to points of the previous query, implies an exact access type and a non-local access sequence. However, searching for a collection of points in the same neighborhood while following a certain path, implies a proximity access type and a local access sequence.

3) Database size

Experiments are often repeated in order to verify a certain phenomena, to determine the statistical behavior of the experiment, or to discover a rare event that occurs only in a small fraction of the experiments. In many cases the results of each experiment can be processed independently. We call each independent part of an experiment a *unit*. An example of an independent unit is a single collision (event) in particle physics, or a single time step calculation of a simulation. It is important to identify such units and to determine their size because they can be processed independently of other units and often in parallel. In addition, if units are small enough they can be processed entirely in main memory, rather than brought piecewise from secondary storage.

Analysis and summarization of experiment data is usually performed over a *collection* of experimental units. The size of a collection is significant because it refers to the quantity of data that analysis queries may need to access. Such queries may select a portion of the collection, or may process the entire set to derive summaries or statistics. A collection may be very large, as is the case with experiments that are run over a period of months because the desired event is rare, because a large number of runs is desirable for statistical analysis, or because extensive parametric studies are desirable.

There is no logical limit to the total amount of data that can be collected by repeating experiments and simulations. The limitations are usually cost and resources. Nevertheless, it is interesting to identify the total amount of data that scientists keep active and available. This category is simply referred to as *total* size. All size figures shown in table 1 are only intended to show order of magnitude.

4) Associated data

The different categories of associated data shown in Table 1: *configuration*, *instrumentation*, *analyzed*, and *summary*, simply indicate whether such data exists for the different example applications. Note that property data is not mentioned since property data is not usually associated with a single experiment, but rather summarizes data over many experiments.

3.2. CHARACTERISTICS OF ASSOCIATED DATA

We chose to emphasize somewhat different characteristics for associated data, because their structure and usage is different from experiment data. The access pattern and size characteristics are similar to those of experiment data, but the identifier characteristics are more diverse. They are described as part of the data modelling characteristics. We also added usage characteristics and non-standard data types.

1) Access pattern

The access pattern characteristics of associated data fall into similar categories as those of experiment data. However, while access patterns of experiment data refer to accessing data points with respect to their identifiers, the access patterns of associated data are with respect to any attributes, whether they are thought of as identifiers or measured data. The reason is that in associated data the concept of an identifier (or category attributes) is not so dominant. For example, when the experiment data of a particle physics experiment are analyzed, the resulting database represents tracks and events rather than the individual data points. The identifiers of the original data points no longer exist in the analyzed data. Instead the tracks and events may be given an identifying number or some combination of the measured values (such as mass and momentum) may be thought of as the identifier.

The categories assigned to access patterns of experiment data above apply to access patterns of associated data as well. However, we found it necessary to add a *partial* access type, because it is common to access associated data (especially analyzed and summary data) by specifying predicates (selection criteria) only on part of the attributes. For example, finding all particles with a mass in a certain range that generated a certain number of sub-particles.

2) Data modelling

The data modelling capabilities chosen here are either common to many examples of associated data, or are included because of their importance. *Geometric* modelling is the capability to describe the geometry of an object (such as an airplane wing), or a collection of objects (such as the position of detectors). The term *entities* refers to the need to distinguish between multiple entities, which is a basic assumption in all database models (such as relational, hierarchical, etc.). There are situations where the concepts of entities are not naturally applicable, such as with summary data (e.g., a co-variance matrix).

The terms *hierarchical* and *networks* refer to relationships between entities. A hierarchical characteristic obviously implies a one-to-many relationship between entities of successive levels of the hierarchy, but also implies the possibility that the identifiers (keys) of higher levels propagate down to lower levels. For example, a particle identifier usually propagates down to its sub-particles level, and is concatenated with the sub-particle identifier to form a unique key. A network characteristic indicates the existence of a many-to-many relationship between entities.

We use the term *generalization* in the sense described in [Smith & Smith 77]. Briefly, it is the capability of describing generically the properties that apply to an entire set of objects. For example, the common properties that describe all analog-to-digital converters of a certain type used in a certain experiment should be described only once. Each individual converter can have its own specific properties, but the generalization capability allows the common properties to be "inherited" by each individual converter.

The existence of multi-dimensional data was explained before in the context of the identifier of experiment data. Although not as common in associated data, the capability to support multi-dimensional data is nevertheless important, especially for analyzed and summary data. We refer to this characteristic as *N-dimensional*. *Meta-data* refers to the information necessary to describe the data. However, the intent here is to emphasize the information that is beyond the usual data definition capability provided by most data management systems. An example of such additional information is the source from which an analyzed database was derived, and the person who derived it.

3) Usage

It is often necessary in the analysis process to change the definition of the database schema, such as to add new attributes (columns) or to calculate new attributes from previous attributes. The ability to support such changes dynamically is referred to here as *schema variation*. Supporting *historical* data implies the maintenance of the history of changes made to the database (not only the latest updated version.) In the implications section we discuss the different aspects of historical data needed for associated data. An important characteristic of a database is its *stability*, i.e. infrequent updates. In physical database design there is

usually a trade off between the efficiency of retrieval and the efficiency of updating. One can take advantage of stable databases to employ more efficient retrieval algorithms in exchange for slower updating.

4) Non-standard data types

The results of the analysis of scientific data are often presented as *graphs*. By *text* we mean not only the usual ability to support character strings of limited size, but also support of unlimited text, such as article abstracts or manual information. The *time series* data type is important in scientific databases (as it is for statistical databases) because special statistical analysis techniques can be applied to time series. The ability to represent the *molecular structure* of materials is a special requirement of scientific data. It cannot be thought of as graphs or images, because it is necessary to be able to refer to the details of the structure, such as "double bonds between certain atoms". We did not include this category in Table 2 because our examples did not have such a requirement, but it is a well known requirement for chemical property data as can be found in many chemical property publications (e.g. The Journal of Chemical Information and Computer Sciences). There is also a need to represent *special symbols* which requires the support of a large character set. By *non-scalar* data type we mean vectors, matrices, and combinations of these. The ability to refer to such objects by name, to refer to particular elements of the objects (such as the i,j element), and to store newly generated non-scalar objects as part of the database is an essential capability for scientific data.

Non-standard data types are discussed in [Hampel & Ries 78].

5) Database size

The size figures shown in Table 2 are intended to show the amount of associated data that is required to support or is generated from a collection (described in section 3.1, part 3) above) of experiment data. The *bytes* figures represent an approximate upper bound, and the *percentage* figures show the approximate size percentage relative to the size of the experiment data collection.

4. EXAMPLES OF SCIENTIFIC DATABASES

In this section we describe a representative scientific application with respect to its characteristics as defined in the previous section. Altogether, we analyzed ten example applications, but because of space limitations they are described elsewhere [Shoshani et al 84]. We have tried to select these examples so that they cover a diverse range of applications. They include simulations, experiments, as well as property data.

In order to have an idea of the kind of applications we analyzed, we describe them briefly below. Then one representative application is described in detail.

The Time Projection Chamber application is designed to record the tracks of sub-particles that result from particle beam collisions. The Limited Track Reconstruction application also deals with sub-particles, but it is designed to collect high resolution measurements on their properties, rather than record their tracks. Hydrodynamics applications are concerned with modelling the flow of fluids, usually using grid methods. Nuclear Magnetic Resonance (NMR) spectroscopy experiments are used to investigate chemical structures. The Heavy Ion Spectrometer application studies the break up process in catastrophic collision involving heavy ions. The passive solar experiment involves the simulation of heat transfer to study the sun energy performance in residence and industry buildings. Turbulent flow studies is another example of hydrodynamics modelling, but rather than using grid methods which require a large number of data points, particle methods are used to model the vorticity of the turbulence. The purpose of the Laser Isotope Separation experiment is to develop a technique for recovering the reusable isotopes from nuclear waste materials.

In addition to the these experiments and simulations applications, two examples of property data were also examined. The function of the Particle Data Group project is to compile particle properties data in a highly evaluated and summarized form. The Nuclear Structure Data project is concerned with recording, evaluating, and tabulating data about the structure of atomic nuclei, and the reactions by which nuclei decay from one state to another.

Next, we describe the Time Projection Chamber application with respect to the characteristics described in section 3.

4.1. Time Projection Chamber

1) Description

The Time Projection Chamber (TPC) is a device used in high energy physics experiments to record the behavior of sub-particles resulting from particle beam collisions. In a typical experiment, two particle beams collide after they are accelerated to very high speeds. Each such collision, called an event, may produce sub-particles that scatter in different directions at different speeds. Often the particles only graze each other and do not produce the sub-particles desired. Because some events are very rare, and because of the need to

be statistically accurate, collision experiments are repeated millions of times.

It is not important here to describe the details of the TPC device, but it is important to understand its operation in order to describe the data generated by it. The TPC is essentially a large cylinder filled with a certain gas. The collisions occur in the center of the cylinder. When particle (or sub-particles) travel through the gas they ionize the gas, leaving "tracks" where they pass. In order to distinguish between positive, negative, and neutral particles, the TPC is subjected to a magnetic field which causes the charged particles to travel in circular patterns which depend on their charge. At the two ends of the cylinder electrostatic fields are applied and cause the ionized tracks to drift to the ends. Special detectors detect the position and time of the drifting tracks, and measure the charge of ions reaching them. From the position of the detector, the x and y coordinates are determined. From the recorded drift time, the z coordinate can later be calculated. The data is collected through special hardware in a binary form onto tapes.

2) The experiment data

2a) Identifier

Each data point of the experiment data consists of a pulse measurement of a certain detector at a certain time. The identifier of each data point consists, therefore, of the position of the detector and the time. The reasons for considering the identifier as having regularity and sparsity are explained next.

The detectors are placed on the two circles at the ends of the TPC cylinder on concentric circles at regular intervals. Because the intervals are regular one can compute the actual x-y position of the detectors by knowing the concentric circle number and the ordinal number of the detector on the circle. The identifier points are said to be ***regular***, because their position can be computed from ordinal numbers, similar to what can be done for a mesh of points.

Readings exist only for the points representing the tracks of the event. Thus, most of the detectors readings are null (in reality, below a very low threshold). Only about one percent of the potential data points have readings. Thus, the identifier is said to be ***sparse***.

There are several techniques that can be used to store identifier data that is regular and sparse. they are discussed in the implications section. the most obvious technique is to throw away the null points and to store the identifier of the non-null points with the data values. This is indeed what is currently done for the TPC experiment.

2b) Access pattern

The first step required before the data can be analyzed is to reconstruct the tracks from the experiment data. The method used is to compute each potential track path, and to verify that data points exist for it. The process of verification involves a search of points along the presumed path. For each such point, the neighboring points are also needed because a pulse has a certain width (for each pulse about 4-6 neighboring data points exist).

The above process exhibits the following access pattern. The access type is ***exact*** match and ***proximity*** search, because for each pulse one looks for a particular point and a collection of points around it. The access sequence is mostly ***non-local***, because each successive collection of points (representing a pulse) are not necessarily close to the previous search. Once a few points are found, the rest of the points are searched along the presumed path. In this case, the access sequence is ***local***, because each successive collection of points would be close to the previous collection.

2c) Size

In a typical six month period, data for about 4 million events are collected. An event is run about once per second, and generates an average of about 28k bytes. A (particularly interesting) large event may generate about 120k bytes. Thus, the total volume of data for a six month period is about 10^{11} bytes, which is stored on about 1350 magnetic tapes. The main difficulty in dealing with such a large volume of data is the mounting and management of tapes for processing. A mass storage system would be most useful for such an application.

Since the data for every event can be analyzed independently from the other events, they can be considered a separate ***unit***. The process of track reconstruction needs only a single unit at a time. However, as discussed later there are other processes that need to be run over a large number of events.

3) The associated data

3a) Configuration data

Although the configuration data does not exist explicitly as a database, it nevertheless exists in the programs analyzing the experiment data. This data corresponds to the description of the physical configuration of the detectors on the TPC device. It consists of mapping information between the identifiers of detectors as stored with each data value and their x-y coordinates. It also includes the mapping of the time measurement to the z coordinate.

3b) Instrumentation data

The instrumentation data is quite extensive and has many components. There is calibration information for each of the 16,000 channels associated with each of the detectors. This information is used to adjust the readings of the detectors. There is other information representing the distortions due to imperfections in the magnetic field, the changes in the electric fields over time, etc. All this information is necessary in order to calibrate the experiment data.

The total amount of instrumentation data is a few megabytes. It is not very large to manage, but it is complex since it contains many components. It is not obvious how to best organize such information in a database management environment.

3c) Analyzed data

The analysis process has many steps that necessitate a number of passes over the experiment data. Each step generates data files that are used in later steps. For example, one of the passes generates histograms over the experiment data. These histograms are used to determine constants for further analysis. A set of (multi-dimensional) histograms is taken over a collection of about 2000 events, and occupies about 400 kbytes. There are about 2000 such sets over the experiment data. These histograms are examples of non-standard data types that require the capability of characterizing and managing an entire data set as a single item.

The final result of this analysis process is to produce summaries about tracks that belong to events. These summaries form the databases that need to be searched for interesting phenomena. Typically, the access type is a *range* search over some particle measures such as mass and momentum. The access sequence is *non-local* since there no a priori correlation between successive queries.

3d) Summary data

Further analysis over the track and event data usually produces graphs and histograms. These data sets need to be managed as non-standard data types.

5. IMPLICATIONS

The implications derived in this section can be best followed by referring to Table 1 for experiment data and Table 2 for associated data. The organization of these tables was designed after the information on the different applications was collected in order to clarify its presentation. However, we believe that these table structures can be used to classify additional applications. Once the appropriate entries are filled for an application, one could quickly draw conclusions on its requirements and the possible data management techniques to support it, along the lines discussed below.

5.1. EXPERIMENT DATA

We discuss the entries of table 1 by referring to its rows because the rows represent observations about each characteristic. The sections below are organized according to the row groups in the tables. The first row labeled "experiment/simulation" is merely to identify whether each example is an experiment or simulation.

1) Identifier

Identifiers in scientific databases are typically multi-dimensional, where the dimensions may be spatial coordinates, time steps, or varying experimental conditions such as temperature or magnetic field changes. An important issue is the efficient storage and access of identifier data which are affected by the regularity, density, and time-variation characteristics.

Identifiers whose dimensions have a *regular* structure are quite common. The main reason is that simpler algorithms can be developed for them, and that the data can be organized in an orderly fashion. The simplest case exists when the configuration of the experiment or simulation forms a multi-dimensional mesh. In such a case there is no need to store the identifiers of the data because the position of each data point can be calculated using the "array linearization" technique mentioned in section 2. Indeed, the array capabilities of programming languages have been used extensively by scientific application. This suggests that an array linearization access method would be most desirable in a scientific data management system. The advantages of such an access method is that it requires no storage for the identifiers and provides a very efficient random access (a simple computation) to the data points.

The situation is more complex when the configuration is not simple, such as representing an airplane wing or the shape of a combustion chamber. In such cases a mesh that covers the entire configuration can be imposed, and all the points outside the configuration boundaries are considered null. This approach introduces a certain level of sparsity in the data points. We will discuss sparsity below.

Other forms of regularity may exist. One is the regular placement of points along some geometric shape, such as concentric circles. Another occurs when two kinds of regular structures co-exist, such as having a finer mesh in certain regions of the configuration. In such cases the mapping algorithm of logical points into

a linear sequence is more complex than array linearization, but they still provide storage savings and more importantly a fast random access to the data points.

As can be seen from Table 1 there are several examples of *irregular* identifiers. At first glance, it seems that identifiers that consist of irregular dimensions, such as the numbers identifying rooms in a building, have to be explicitly stored with each corresponding data value. Such an approach wastes space since each dimension value has to be repeatedly stored with the data. Rather, the irregular dimensions can be enumerated and stored only once. Thereafter, the identifiers can be calculated using array linearization over the enumerations. Irregular dimensions are most common in statistical databases (such as state, race, sex, and cause of death for mortality data), where the enumeration of each dimension and array linearization over them is a most effective method.

Data *sparsity* means that only a fraction of the points in the full cross product of the dimensions have actual values associated with them. There are basically two options: either to store the identifiers of the valid data points, or to compress out the non-valid (null) data points. Compression methods, such as run length encoding (which introduce a count into the data stream in place of each sequence of null points) can be quite effective, especially when the null points are clustered to form long sequences. However, such compression methods require sequential scanning of the data in order to select a particular point randomly. Indexing methods require too much space for large databases and may be prohibitive.

In [Eggers & Shoshani 80] a compression technique, called header compression, which provides fast (logarithmic) access was proposed for statistical databases. It basically organizes the run length counts into a separate header, in such a way that the header can be searched in logarithmic time with respect to the number of counts. This technique can be applicable for sparse scientific data as well, since it can be used effectively with multi-dimensional data.

Time varying applications are not as common as other applications, but they represent an important class of modelling techniques. When the identifiers are *time varying* there is no choice but to store them, since they change from one time step to the next. In the case that the data is also regular, there is an additional requirement that the original relationship between the points is maintained. To see this point, one can imagine a mesh of points connected by rubber strings. The entire structure can then be stretched and compressed in successive time steps. The maintenance of these relationships can be achieved with techniques applicable to regular data. When data is irregular and time varying, the relationship between the data points changes from one time step to the next, and has to be deduced from the stored identifiers.

2) Access pattern

From Table 1 it can be seen that the access types of *exact* match and *proximity* search are important. Exact match implies, in general, the need to access specific data points randomly. To accommodate such a requirement some kind of indexing or hashing technique is required. Fortunately, one can take advantage of the multi-dimensionality of the data. The mapping of multi-dimensional space to linear space discussed above (e.g. array linearization) provides a key-to-address mapping that is equivalent to hashing. In addition, some multi-dimensional to linear mappings provide advantages for proximity search as discussed below.

To support proximity search it is necessary to preserve logical locality in physical storage. That is, when points are logically close to each other in the multi-dimensional space, it is desirable that they are physically close in physical space, so that they can be brought into memory from secondary storage with a minimum number of accesses. This suggests the organization of physical storage into cells along the dimensions of the identifier. The data points within a cell will satisfy the proximity requirement. For elements on the borders of cells it is necessary to access adjacent cells, and therefore the placement of cells in physical storage is also important. The mapping of multi-dimensional space to linear space mentioned above works well with such a cellular organization because it does not disturb the logical proximity of the data points. An arbitrary hash mapping would place data points into cells (buckets) which would not necessarily preserve logical proximity. The optimal partitioning of cells, especially in the case of sparse data, is an interesting problem that should be further investigated.

The *range* access type does not seem to be as important. Nevertheless, the cell organization should benefit range access on the dimensions of the cells.

Referring again to Table 1 it seems that *local* access sequence is also important. The cell organization is also helpful here because local points are likely to be in the same cell. The question of how to organize the cells arises here again. If the paths of local access sequences are known or predictable, then the cells should be organized along these paths. The benefits of such ideas need to be investigated.

Non-local access sequence is not as prevalent as local access sequence. However, it can be supported well with cell organization. The reason is that it complements the requirements of exact match, since it implies the need for a random access of the data points. *Linear*

access sequence conflicts with the idea of a cell organization, because the linear sequencing of the data is broken. However, it does not seem to be an important requirement. If data was organized "linearly" to accommodate this requirement, then proximity search and local access sequence will be performed less efficiently.

An *arbitrary* access sequence is quite common. It usually implies that the entire data set needs to be processed, and that the order of points is irrelevant. This suggests that parallel processing can be performed over the data. This only complements the cell organization approach, since the cells could be placed on parallel devices for parallel processing.

In summary, it seems that the cell approach is most desirable since it accommodates the most important requirements. The organization of cells should be along the dimensions of the identifier, since they preserve logical locality. The approach of mapping the multi-dimensional space into linear space complements this cell organization. There are several papers that discuss the organization of data into cells [e.g. Nievergelt et al 84]. However, the access requirement mentioned here, such as proximity search and local access sequence were not explicitly addressed.

3) Size

The most important observation that can be made from the size figures in Table 1, is that although scientific databases are large, they can often be partitioned into small independent *units*. The units are small enough that much of the processing can be done in main memory. In general, experimental units can be processed in parallel, since they are independent of each other. Simulation units (time steps), on the other hand, usually follow each other in sequence. Note that simulation units are typically larger than experimental units.

Unit processing is only one part of the analysis process. Other types of processing need to search and access entire *collections*. As can be seen from the collection figures in Table 1, some collections are so large that they cannot be practically stored on magnetic disks. In such cases, the data is currently stored on tapes and the mounting of those tapes becomes a major problem. Current solutions are to process the data sequentially once, to collect interesting subsets, or to break the data into redundant smaller sections. It is obvious that larger secondary storage devices (such as optical disks) could be helpful.

4) Associated data

Associated data is discussed in the next section. The different types of associated data were included in Table 1 in order to point out their importance and prevalence. Nearly all applications have all types of associated data. The obvious exception is that simulations do not have instrumentation data.

5.2. ASSOCIATED DATA

Table 2 summarizes our observations on the different types of associated data. We could discuss these observations by row for each class of characteristics or by column for each type of associated data. A close observation of table 2 reveals that there are many similarities between the configuration and instrumentation columns, and between analyzed data and summary data columns. This is not very surprising since these two groups represent support data and generated data and should have similar characteristics. In fact, early on we did not make this finer distinction, but later we found that it helped sorting out the different aspects of scientific data.

Accordingly, we will discuss characteristics in Table 2 in three parts: the support data (configuration and instrumentation data), the generated data (analyzed and summary data), and property data.

1) Support data

The access type for support data is mostly exact match. A typical access involves finding a particular configuration point and the particular instrument associated with it. Proximity search is sometimes needed. For example, if a certain instrument failed, the configuration data may be consulted to find the instruments in neighboring locations. The access sequence is mostly non-local, which indicates that successive queries are unrelated. Thus, the access requirement for support data is mainly random access.

The data modelling requirements are fairly conventional, i.e. modelling of entities that have hierarchical or network relationships. The relationships between the different instruments and detectors are part of the configuration data. Generalization is an important modelling tool for instruments, as generic information can be represented once and inherited by each particular instrument in that class.

An important exception to the conventional modelling requirements mentioned above is geometric modelling of configuration data. In many examples the geometry is quite regular and could probably be modelled with simple types (points, lines, circles, etc.). However, geometric shapes may be complex enough to require special modelling techniques similar to those required in engineering databases [Lorie 82].

Another major requirement is for the support of historical data. Instrumentation data change continuously over time, and the entire history of changes has to

be recorded. In addition, logs of the operation, such as when an instrument failed, who was in charge at the time, etc. need also be recorded. The time element can be thought of as another dimension orthogonal to the structure of the database. It requires special storage techniques and special operators such as "after" and "during". Several recent works have dealt with this topic [e.g. Anderson 81, Bolour et al 82]. The history of configuration data changes also needs to be recorded, but not as often as instrumentation data because they usually occur only between experiments.

Support data may have some text that describe procedural instructions or configuration descriptions. Instrument data are usually polled at regular time intervals, and could benefit from a time series data type. The size of the data is relatively small, and constitutes only about 1% of the experiment data.

In conclusion, we believe that support data can be managed for the most part with conventional data management techniques. The databases are relatively small. The requirement for random access can be accommodated with conventional indexing or hashing methods. The two most important exceptions that require special attention are historical data support and geometric modelling.

2) Generated data

The access pattern of generated data is similar to statistical databases. That is, it is mostly range and partial match queries. As with statistical databases, the generated data is repeatedly analyzed in order to discover patterns, statistical behavior, or a rare event. Many subsets are generated and need to be kept track of. The access sequence is mostly non-local, although locality exists when analysts refine their queries. From time to time an entire set of analyzed data is processed to generate summaries. This is indicated as an arbitrary access sequence in Table 2.

The most prominent data modelling characteristic is that generated data is multi-dimensional. Unlike experiment data where the dimensions are mostly spatial coordinates, the dimensions of generated data are the properties of the data (e.g. charge, temperature, mass). Thus, the number of dimensions can be in the order of ten, which presents a special challenge for its efficient support. In some instances it is useful to view analyzed data as entities and hierarchical relationships (for example, events and their corresponding subparticles).

Another important modelling requirement is for meta-data. The requirements of meta-data management include data definition facilities not only for field descriptors (such as type, size, and acronym), but also the description of the origin of the data, how it was collected, when it was generated or modified, and the identity of the person responsible for its collection. Facilities to describe complex data types such as times series, matrices, and multi-dimensional categorical data are also needed.

It is necessary to organize and manage meta-data, just as is the case with data. One should be able to retrieve and search meta-data, index keywords, and browse through the meta-data structures. A system that supports such operations for statistical databases is described in [Chan & Shoshani 81].

Meta-data is also necessary for keeping track of the different subsets produced, dates of their creation, methods used, etc. The management of subsets also requires that their historical aspects are maintained. It is necessary to record and maintain the ancestors of each subset produced. The analysis process, similar to statistical analysis, can be modelled as a tree or a directed graph structure. The analyst can generate subsets, observe their patterns, and choose to go back to a previous set and follow another path of analysis. The above requirements are similar to many aspects of the meta-data management for statistical databases [McCarthy 82].

It is often useful in the analysis process to add new fields to the database or to compute new fields from other fields. This is referred to in Table 2 as schema variation. However, except for such additions during the analysis process, the generated databases are quite stable. The support of non-standard data types is most important. Generated data can be expressed as graphs, vectors, matrices, and time series. Finally, the size of the data is substantial, and although it is small enough to fit on disks, it is sufficiently large to benefit from data management techniques that minimize disk storage and access time. The total amount of generated data may be of the same order of magnitude as the experiment data it was derived from, because a large number of subsets are usually produced.

In conclusion, generated data have many characteristics in common with statistical databases. We believe that special techniques for the management of multi-dimensional data developed for statistical databases could be applied to support analyzed data.

3) Property data

Since property data is a summary over many experiments and contain general knowledge of a subject area, it has characteristics more akin to bibliographic

databases. However, in addition to managing bibliographic data on books, articles, authors, etc. they contain summaries of information extracted from the articles. Many property databases are presently only available in periodic publications.

Different access types are needed. An exact match may be required to locate a specific entry. A range query (which may be partial) could be used to find a desirable subset of the data. A proximity search will locate entries with properties as close as possible to the specified parameters. The access sequence is usually non-local. Linear access sequence is needed for generating the periodic reports, or some other requested report. In the examples that we observed it is not possible to issue ad-hoc queries or to browse the database for information since the databases are not available on-line. An on-line version should provide such facilities.

The databases are organized logically as entities and mostly hierarchical relationships between them (for example, particles with a certain mass at the top level of the hierarchy, and their derivatives at lower levels). There are some network relationships (permitting a many-to-many association between the entities) such as the relationship between papers and particles. A single paper may describe many particles and a particle may be described in many papers.

Property databases contain graphs and text which complicate their management a great deal. They also have to have a representation for special symbols that are unique to the scientific field. The total size of the example databases that we observed is not very large and can fit on disk storage. However, property databases can be quite large, as is the case with chemical property databases.

In conclusion, the main difficulties observed in property databases stem from the diversity of data. This is probably the reason that only special purpose systems have been developed for the different disciplines. Conventional data management systems do not support the combination of numeric, character, text, and graphs and image data.

6. SUMMARY

In this paper we examined the data management requirements and typical usage of several scientific applications. The data used and generated by these applications was classified into types, and a list of properties for describing their characteristics was developed. Different applications were then described in terms of this list of properties. A summary of properties of the different types of scientific data provided the basis for inferring desirable database management techniques for scientific databases.

Some of the more important conclusions are:

(1) Multi-dimensional data are prevalent in scientific databases. Methods for efficiently managing, accessing, and compressing multi-dimensional data are desirable.

(2) Scientific databases are frequently accessed via proximity searches and successive queries often exhibit locality of reference. Techniques of partitioning the data into cells (or grids) along the coordinates of its dimensions seem to be the most promising for efficiently supporting these needs.

(3) Although scientific databases are usually very large, they can be often partitioned into small independent units during early data reduction. This implies that parallel processing can be applied.

(4) Scientific databases include a variety of support data that describe instruments and the configuration of experiments. Often this data is not explicitly organized but rather made part of application programs, a practice that tends to cause many difficulties. The requirements of such support data can be handled for the most part with conventional database techniques, but need to be integrated with the data that result from experiments. Some configuration data need special capabilities found in engineering database systems.

(5) The analysis of scientific data generates many summary data sets which need to be managed. Special techniques for handling analyzed data and summary data are required in order to manage their metadata, to keep track of numerous data sets, and to handle non-scalar data types (such as vectors and matrices).

(6) Historical aspects of scientific databases are important. They range from time series of the measured data, to logs of instrument variation over time, to the historical sequence of generating different summaries of the data.

(7) There are many aspects of scientific databases that are similar to statistical databases; in particular, supporting the multi-dimensional aspects of the data and the handling of summary data.

Acknowledgements

We would like to thank the many scientists who contributed their time and effort to explain their scientific applications, current processing techniques, open problems, and future needs. The list includes: Brandt Anderson, Marsha Berger, Mark Bronson, Eddie Browne, Bill Carroll, Phil Colella, Janis Dairiki, Margaret Garnjost, Van Jacobson, Rowland Johnson, Stu Loken, Gerry Lynch, Creve Maples, Chuck McParland, Michael

Munowitz, Alex Pines, Al Rittenberg, Jeff Saltzman, Jamie Sethian, Joe Sventek, Tom Tripp, Tom Webster.

We also like to thank our colleagues Paul Chan, John McCarthy, and Doron Rotem, for many fruitful discussions. Special thanks are due to Meri Jones for her text processing help.

This work was supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under contract number DE-AC03-763F00098.

REFERENCES

[Anderson 81]
Anderson, T.L., *The Database Semantics of Time*, Ph.D Dissertation, Dept. of Computer Science, Univ. of Washington, 1981.

[Bell 83]
Bell, Jean L., *Data Structures for Scientific Simulation*, Ph.D. dissertation, Univ. of Colorado, Dept. of Computer Science, 1983

[Bolour et al 82]
Bolour, A., Anderson, T.L., Dekeyser, L.J., Wong, H.K.T., The Role of Time in Information Processing: A Survey, *ACM-SIGMOD Record*, 12, 3, 1982, pp. 27-50.

[Chan & Shoshani 81] Chan, P., Shoshani, A., Subject: A Directory driven System for Organizing and Accessing Large Statistical Databases, *Proceedings of the International Conference on Very Large Data Base (VLDB)*, 1980, pp. 553-563.

[Eggers & Shoshani 80]
Eggers, S. J., Shoshani, A., Efficient Access of Compressed Data, *Proceedings of the International Conference on Very Large Databases*, 6, 1980, pp. 205-211.

[Hampel & Ries 78]
Hampel, M., Ries, D. R., Requirements for the Design of a Scientific Data Base Management Systems, Special Report on *Generalized Data Management Systems and Scientific Information*, OECD Nuclear Energy Agency, Paris, 1978, pp. 111-131.

[Lorie 82]
Lorie, R.A., Issues in Databases for Design Applications, *File Structures and Data Bases for CAD*, J. Encarnacao and F.L. Krause (eds), North-Holland, 1982.

[McCarthy 82]
McCarthy J., Meta-data Management for Large Statistical Databases, *Proceedings of the International Conference on Very Large Data Base (VLDB)*, 1982.

[Nievergelt et al 84]
Nievergelt J., Hinterberger H., Sevcik K.C., The Grid File: An Adaptable, Symmetric Multikey File Structure, *ACM Transactions on Database Systems 9*, 1, March 1984, pp. 38-71.

[Smith & Smith 77] Smith, J. M., and Smith, D. C. P., Database abstractions: aggregation and generalization, *ACM Transactions on Database Systems 2*, 2, June 1977, pp. 105-133.

[Shoshani 82]
Shoshani, A., Statistical Databases: Characteristics, Problems, and Some Solutions, *Proceedings of the 8th International Conference on Very Large Data Bases (VLDB)*, 1982, pp.208-222.

[Shoshani et al 84]
Shoshani, A., Olken, F., Wong, H.K.T., Characteristics of Scientific Databases, Lawrence Berkeley Laboratory document number LBL-17582, March 1984.

Characteristics	Time Projection Chamber	Limited Track Reconstruction	Hydro-Dynamics	NMR Spectroscopy	Heavy Ion	Passive Solar	Turbulent Flow (Vortex)	Laser Isotope Separation
EXPERIMENT / SIMULATION	E	E	S	E	E	S	S	E
IDENTIFIER (KEY)								
Regular	•	•	•	•				
Dense			•	•	•	•	•	•
Time Variation			(•)				•	
ACCESS PATTERN								
Access Type								
Exact	•		•	•		•		
Range		(•)	(•)				(•)	(•)
Proximity	•		•		•	•	•	(•)
Access Sequence								
Local	(•)		(•)	•			(•)	(•)
Non-Local	(•)				•			
Linear			(•)					
Arbitrary	•	•	(•)		•	(•)		
SIZE (bytes)								
Per Unit	10^{4-5}	10^{3-4}	10^{7}	--	10^{3-4}	10^{3-4}	10^{6}	--
Per Collection	10^{11}	10^{10}	10^{8-9}	10^{5-7}	10^{10-11}	10^{6-7}	10^{8}	10^{7-8}
Total	10^{12}	10^{11}	10^{11-12}	10^{8-10}	10^{12}	10^{9-10}	10^{10}	10^{10}
ASSOCIATED DATA								
Configuration	•	•	•	•	•	•	•	•
Instrumentation	•	•		•	•			•
Analyzed	•	•	•	•	•		•	•
Summary	•	•	•	•	•	•		•

Table 1: Summary of Characteristics of Experiment Data

• - *applies often*
(•) - applies sometimes

Characteristics	Configuration	Instrumentation	Analyzed Data	Summary Data	Property Data
ACCESS PATTERN					
Access Type					
Exact	•	•			•
Range			•	•	•
Proximity	(•)				(•)
Partial			•	•	•
Access Sequence					
Local	(•)	(•)	(•)		
Non-Local	•	•	•	•	•
Linear					•
Arbitrary			(•)		
DATA MODELING					
Geometric	•				
Entities	(•)	•	(•)		•
Hierarchical	(•)		(•)		•
Network	(•)				(•)
Generalization		•			(•)
N-Dimensional			•	•	
Meta-Data			•	•	
USAGE					
Schema Variation			•	•	
Historical	(•)	•	•	•	
Stability	•		•	•	
NON-STANDARD DATA TYPES					
Graphs			(•)	•	•
Text	(•)	(•)			•
Time Series		•	(•)	(•)	
Special Symbols					•
Non-Scalar			•	•	
SIZE					
Bytes	10^{6}	10^{7}	10^{8-9}	10^{6}	10^{7-8}
Percentage	1%	1%	50-100%	1-10%	

Table 2: Summary of Characteristics of Associated Data