MASTER

DISTRIBUTED DATA ANALYSIS IN A MOBILE REAL TIME AND MINICOMPUTER NETWORK ENVIRONMENT*

James J. Thomas
Gerald A. Sandness
James O. Schroeder
Arthur J. Boland

Battelle-Northwest
Richland, Washington 99352

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or 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 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. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Abstract

The system design and analysis processes involved in collecting data from a mobile laboratory to a finished analysis are presented. This involves real time data collection on a small mobile cart, an intermediate analysis station in a nearby mobile van, and the final analysis performed through a minicomputer network of facilities.

An iterative approach to data collection and review was established to provide timely analysis and verification of incoming data. This requires a highly interactive user-oriented environment in the field. After the data has been collected and preliminary analysis completed, it can be integrated into a larger analysis system through a minicomputer network. This also allows for utilization of high cost equipment that cannot be placed in the field.

A single hardware interface "BATNET" (Battelle Network) is used to perform all data transfers either via radio (cart-to-van) or external network connections (van-to-computer network). The key software components described are: a self describing data base system utilizing simple relational data base concepts, a simple command language allowing analysis and investigation in the field, and a simple preview system for intensity modulated data.

eb
DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

*Prepared for the U.S. Department of Energy, formerly the Energy Research and Development Administration, under Contract EY-76-C-06-1830.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or 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 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. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

The following pages are an exact
representation of what is in the original
document folder.

INTRODUCTION

Interactive data analysis is the technique for exploring data usually without a preconceived structure to the solution process. The decisions leading to the next solution step are based on each previous step with the man-machine interaction being a key element. Most implementations of interactive data analysis systems have been aimed at large scale machines (1,2,5). This paper describes an effort to provide interactive data analysis in a medium scale minicomputer acting as an intermediate step in a distributed processing configuration. The key design goals were to provide a user level command language for acquired real time data, an efficient data management system independent of data semantics, a self describing data structure to allow for transportability of data, and graphical preview system.

In essence, this effort attempts to apply the basics of several advancing technologies to a real time mobile data analysis system.

The emphasis of this paper is on the design objectives and decisions relating to the data structure and user level command language. The areas discussed are computational design philosophy of the distributed system, the facilities support, the data structure, the command language, and the intensity modulated contour previewing.

COMPUTATIONAL DESIGN PHILOSOPHY

A multi-station approach was taken to implement this system; a station being a computer facility designed to perform a limited number of tasks well. The stations involved ranged in capability from a microcomputer controlling the data acquisition to a large minicomputer network center. Figure I shows a conceptual view of the station design philosophy.

FIGURE I

The micro station is designed to control data acquisition involving several scientific instruments. The minicomputer station is intended to provide a high degree of interaction and immediate feedback. A key component in the design of this station was to provide for monitoring incoming data, checking for validity and

reasonableness, and directing further investigation. This station should also allow for combining intermediate results to signal a retake data condition if necessary. The large scale minicomputer station and network provides extensive analysis and report quality graphics devices.

FACILITIES SUPPORT

The facilities for the three stations involved in this distributed data collection and analysis system are illustrated in Figure II station:

FIGURE II

The last Station C is actually a minicomputer network (3) with each mini having a specialized set of peripherals, i.e. DICOMED COLOR IMAGE RECORDER. Station A is located on a nonmagnetic, 3 wheel, self-contained land rover. Station B is located in a nearby mobile van. Station C is located at the Battelle-Northwest (BNW) research complex.

The microprocessor is used to gather data from the on-board instruments, perform limited amounts of signal averaging, and organize the data in a message packet to be sent to the analysis station. Instruments like a RADAR system collect ANALOG data which are converted to digital data via an A/D converter. The digital data is then transmitted to the analysis station using the BATNET telemetry system. The BATNET (3) interface is used to transmit or receive data in bit serial, pulse code modulated (PCM) format. A BATNET message consists of a start bit, 16 bit words, and a 16 bit cyclic check character. The transmission rate is software selectable from 11,500 bits/sec to 3.0M bits/sec. For the telemetry system we have slowed this down to 50K bits/sec due to limitations in the transceivers. Communication between the BATNET interface and a host computer takes place via direct memory access and interrupt driven low-level software. The data is double buffered in the microprocessor so that the data collection and transmission can proceed simultaneously. The analysis station and microprocessor station must synchronize data transmit and receive operations so that the analysis station has enabled the BATNET interface in the receiver mode prior to data transmission. The BATNET interface detects data transmission errors by examining the checksum word appended to the message buffer. If errors are detected, the receiving task is notified for corrective action.

The information is then gathered in the analysis station. The processing and analysis techniques are described in the remainder of this paper. The BATNET interface to transfer data to and between nodes of the minicomputer network is the same physical hardware that is used to transmit data from Stations A and B. An external cable on the mobile van allows us to drive the van close to the BNW Laboratory Complex and directly connect into the high speed minicomputer network.

DATA STRUCTURE

The design philosophy for the data structure was to provide a data structure with the following properties:

- Simple to understand
- Modifiable without programming
- Accessible through simple command query system
- Low overhead
- Shareable among many computer systems

To achieve the above properties the following decisions were made:

- Use a simple relational data base structure.
- Utilize a self describing data structure. That is, the description of the file as contained in the file.
- Design a basic set of access utilities either callable via program or invokable via a command language.
- Design a command language that has relational query and data manipulation capabilities with simple short command format.
- Experiment with combining graphics, analysis and data base management under one system.

Self describing data formats (4,6,7) is not a new concept. Having the file description in the file allows each node on a distributed processing system to provide its own Data Base Management System (DBMS). The technique is called the integrated approach to distributed data bases (8).

Several generalized systems have also been designed for relational models, which usually run on large computer configurations (1,2,5). Often the command structure is complex and discourages the problem solver from using the system. Therefore, this experiment was undertaken to incorporate a simple short command language with the resultant data management system called the "Interactive Data Analysis" system (IDA).

In order to provide an easy to understand concept of the relational data model, the terminology of rows and columns will be used instead of the domain and range spaces (1).

The basic data structure involved is a collection of files each containing its own description, both syntactic (6) and limited semantic properties.

The data description, physically located in the first few blocks of the file, contains information such as the logical number of rows, physical number of rows, column names, column length, and column type. When a file becomes active for reference, its data definition is mapped into memory allowing general file characteristic queries without further file access. In most applications there is a need to log semantic and often unique data properties. To accommodate this, a standard means of being nonstandard was provided in this data description. Other technologies have found this a viable technique to ensure a base interchange format (9,10).

To create a file involves defining the file structure in an ASCII, editable, file-description file. This has the advantage of allowing the user to directly edit and create new file structures without program modification. This also has the disadvantage that the problem solver must have a preconceived knowledge of the data model before a new file creation.

The file-description file consists of a list of column names, types (integer, floating, character...) and length. In many applications it was observed that the column name only varied slightly while most of the characters remained constant. In order to enhance the user interface, a facility was designed to allow column names to be repeated with a three digit instance number concatenated in the name. The following example describes a file that contains 124 logical columns with the last set of column names being RAD001, RAD002,...RAD120.

<u>COLUMN NAME</u>	<u>LENGTH BYTES</u>	<u>TYPE</u>
XPOS	2	I
YPOS	2	I
MAGN01	2	I
MAGN02	2	I
RAD001	1	I
DUPL	119	

When a file is created the file-description file is referenced and a condensed form is stored at the beginning of the file. As the file is modified, the current file specifications are also logged in the file.

The key internal data structures, when a file is active for modifications, include a column name table and a description table for each file.

COLUMN
NAME
TABLE

NAME1
NAME2
NAME3
•
•
•

COLUMN
DESCRIPTION
TABLE

POSITION	L	T
"	"	"
"	"	"

POSITION is the physical row position, L is the length in bytes, and T is the data type.

Utility access routines such as get-an-item or process-a-relational-expression use these tables to locate and collect data items. Several other structures are required to control single and multiple file operations. These will not be addressed in the scope of this paper. Other facilities and structures that allow query responses will be discussed under command languages.

COMMAND LANGUAGE

The command language has been designed to allow nonprocedural user-oriented data queries. Several other command languages have been designed with similar goals (1,2,11,12). A major design goal was to keep the commands as short as possible while still allowing free form commands. The command language allows for the following facilities:

- Creating and closing new data files
- Activating/deactivating an existing file
- Listing all data file properties; names, sizes, fields,... etc.
- Selection commands for rows, columns, both by absolute row numbers and column names and by data item contents
- Subsetting, listing, merging, and storing files based on selected results
- Graph selected results

The complete command syntax will not be given but a sample sequence should illustrate the command format and usage.

ACTIVATE RADFIL	;activate the file RADFIL	*
COLSELECT XPOS,YPOS	;select 2 columns	
LIST RADFIL	;list the file under selected columns	
ROWSELECT 1...1000	;select the first 1000 rows	**
COLSELECT XPOS,YPOS,RAD001	...RAD120	
	;select X & Y position and 120 radar values**	
GRAPH RADFIL	;graph file	
CONDITIONS FOR XPOS>10 & YPOS<1000		
	;set conditions to select all rows with	
	;XPOS>10 and YPOS<1000	***
CREATE NEWFIL	;create a new file called NEWFIL	
SUBSET RADFIL INTO NEWFIL	;subset operation based on conditions	
GRAPH NEWFIL	;graph the new file	
CLOSE NEWFIL	;close the new file	
DEACTIVATE RADFIL	;deactivate RADFIL	
END	;terminate analysis session	

***Any complex relation may be specified involving arithmetic and logical relations.

**One of the features allowed by the command language is the positional selection of data items in relation to other logical items in the users data model. This feature, while simple in concept, provides a great deal of flexibility with little typing.

*All function words like ACTIVATE, SUBSET may be abbreviated to the first three characters.

Combining or reordering commands is allowed. An example would be:

SUBSET RADFIL INTO NEWFIL FOR XPOS>10 & YPOS<100

or

FOR XPOS>10 & YPOS<100 SUBSET RADFIL INTO NEWFIL

This allows for complex commands and is not recommended for daily use. Our initial experience shows that a user will type a set of relations, then verify the desired relations before performing a sub-set operation that affects the file structure. The short commands also seem to prevent erroneous file modifications.

Each command is parsed into an application independent definition. The parsed definition includes a set of tables consisting of functional or operational information, source file information, target file information, and relational and conditional tables. All of the above information is stored in tables having the same format, table length, current-high-water-mark or next available slot, and the command syntax analysis. The relational and conditional tables are divided to provide the absolute row selection, absolute column selection, and a polish stack representation for the conditional relations. Thus each command processor is a semantic interpreter having available the above syntax analysis tables.

INTENSITY MODULATED DATA PREVIEWING

A critical part of this system was to provide medium quality (low cost) as compared to high quality (high cost) graphic output in a semi-real time environment. To do that, a quick image facility was designed. This allows intensity modulated data like magnometer or level slicing on RADAR data over a grid space to be plotted directly with no immediate storage or processing steps. To do this, a 200 DPI electrostatic printer/plotter was selected. Each data item is simulated by an 8x8 matrix with varying densities of dots to simulate intensity modulation. Each plot of items is scaled to determine a number of levels. Then the data to be plotted is sent to a utility routine that detects and highlights level changes creating a contoured plot. This process allows fast plotting on a medium quality system. Obviously color would provide another dimension and is planned for future expansions. Since gradient changes between levels are not as distinguishable as desired, a number plotting scheme was designed to allow plotting 99 levels. This scheme again produced medium quality plots which are more than adequate for the in-the-field results. A few examples are shown in the Appendix.

It is important to note that these raster intensity modulated plots are generated in semi-real time without use of secondard storage as intermediate processing steps usually found in raster image output.

CONCLUSIONS

The described system is operational and has about three months in-the-field experience. The data base management system, while simple in concept and implementation, has proven to be a viable technique for medium scale analysis systems. The fact that the data contains its own description did allow the analysis of the data to be split between a nearby minicomputer network and the analysis station. The user oriented command language is a must as the problem solver or scientist will be the individual controlling the system during experimental data gathering and analysis.

The most difficult portion to design on the interface between graphics and the extensible mathematical operations. This is a definite future research effort.

REFERENCES

1. J. D. Joyce and N. N. Oliver, REGIS - A Relational Information System with Graphics and Statistics, A FIPS National Computer Conference Proceedings, Vol. 45, 1976, 839-844.
2. N. N. Oliver and J. D. Joyce, Performance Monitor for a Relational Information System, ACM Conference Proceedings, October 1976, pp. 328-333.
3. L. H. Gerhardstein, et al., The Pacific Northwest Laboratory Mini-computer Network, Proceedings of Berkeley Workshop on Distributed Data Management on Computer Networks, August 1978.
4. A. A. Brooks, An Extension of the ANSI 239.2 Standard to General Information Exchange, Proceedings of Workshop on Distributed Data Management and Computer Networks, May 1976.
5. Codd, E. F., A Relational Model of Data for Large Shared Data Bases, Comm. ACM 13, June 1970, pp. 377-387.
6. CODASYL, Data Base Task Group Report, April 1971.
7. A. Shoshani, On the Importance of Common Standards for Logical Structure, Data Formats, and Query Languages, Proceedings of the Berkeley Workshop on Distributed Data Management and Computer Networks, May 1976.
8. A. Shoshani, Data Sharing in Computer Networks, WESCON Conference, September 1972.
9. R. F. Sproull, Network Graphics Isn't Working, Proceedings of the Berkeley Workshop on Distributed Data Management and Computer Networks, May 1976.
10. M. Marcotty and H. Schutz, The Systems Language MALUS, Software Practice and Experience, Vol. 4, 79-90 (1974).
11. N. McDonald and M. Stonebraker, CUPID - The Friendly Query Language, Laboratory Memorandum ERL-MY87 University of California, Berkeley, CA, October 1974.
12. G. H. Held, M. Stonebraker and E. Wong, INGRES - A Relational Data Base System, A FIPS Conference Proceedings National Computer Conference, Vol. 44, 1975.