IBM CAMBRIDGE SCIENTIFIC CENTER
G320-2125, December 1978

SPARSITY AND APL

R. I. FRANK

1976 IBM CAMBRIDGE SCIENTIFIC CENTER REPORTS

320-2110 April 1976
J. RUBIN, A Resource Allocation and Job Scheduling Model (22 p.)

320-2111 April 1976
Y. BARD, A Characterization of VM/370 Workload (40 p.)

320-2112 May 1976
T. MORE, JR., Types and Prototypes in a Theory of Arrays (65 p.)

320-2113 May 1976
T. MORE, JR., On The Composition of Array-Theoretic Operations (62 p.)

320-2114 May 1976
R. I. FRANK, Design Considerations in an Experimental Interactive APL Quadratic/Linear Programming Workspace (28 p.)

320-2115 May 1976
S. G. GREENBERG, High Level Data Management for the Problem Solver (14 p.)

320-2116 December 1976
W. P. FISCHER, The Heating Oil Consumption of One-Family Houses in Eastern Massachusetts (66 p.)

320-2117 December 1976
W. P. FISCHER, A Note on Residential Heating Oil Inventory Policies (31 p.)

320-2118 December 1976
W. P. FISCHER, Short Term Changes in the Residential Consumption of Heating Oil (25 p.)

320-2119 December 1976
W. P. FISCHER, Short Term Residential Demand for Heating Energy in New England (45 p.)

320-2120 December 1976
M. SCHATZOFF, System Performance, Measurement and Tuning (37 p.)

320-2275 December 1976
Compiled by LILES SMITH, Abstracts of Cambridge Scientific Center Reports (113 p.)

320-3175 May 1976
Abstracts of Philadelphia Scientific Center Reports (126 p.)

1977 IBM CAMBRIDGE SCIENTIFIC CENTER REPORTS

320-2121 June 1977
Y. BARD, An Analytic Model of the VM/370 System (43 p.)

1978 IBM CAMBRIDGE SCIENTIFIC CENTER REPORTS

320-2122 July 1978
P. BERKE, Tables, Files and Relations in Array Theory (27 p.)

320-2123 July 1978
P. BERKE, Data Design with Array Theory (33 p.)

G320-2124, November 1978
Y. BARD, Some Extensions to Multiclass Queueing (27 p.)

G320-2125, December 1978
R. I. FRANK, Sparsity and APL, (12 p.)

G320-2126, December 1978
R. I. FRANK, The Differential Geometry of the C^2 Unconstrained Optimization Surface, (21 p.)

The availability of reports is correct as of the printing date of this report.

\+ Appeared in an outside publication. Not available in Scientific Center report form. Please refer to the list of outside publications on the inside back cover for availability of reprints.

• Copies of report are no longer available from the Scientific Center.

IBM CAMBRIDGE SCIENTIFIC CENTER REPORT NO. G320-2125, DECEMBER 1978

SPARSITY AND APL

R. I. Frank

IBM Corporation
Cambridge Scientific Center
545 Technology Square
Cambridge, Massachusetts 01239

ABSTRACT

A definition of Sparse APL is proposed. A review of Sparse Matrix Research is presented and related to the current APL language and systems. Presented next is a discussion of the possible extensions of classic Sparse Matrix methods that would be required in a "Sparse Array" implementation of APL. This includes a discussion of a suggested research direction. Some relationships of Sparse APL to other fields outside of APL are mentioned. Finally, a list of open questions is interposed for anyone interested in pursuing some of these questions.

SPARSITY AND APL

1. SPARSE APL

Sparse APL would be an APL language unchanged from its natural course of evolution but implemented in a system that would not use storage space or operation executions when they are "unnecessary".

This ideal system is a goal, not a reality. There are many problems in design and implementation yet to be overcome, not the least of which is the precise specification of when an item must "necessarily" be stored or be a part of an operation execution. Also, it is not certain that the ability to handle "Sparse" data can be arrived at without some new linguistic facilities, especially during input and output.

"Sparse" data are data that can be specified with less information than that required to store each element in its proper array position. For example, the vector of the first million integers does not require one hundred storage locations and certainly not one million.

In order to be well defined, this definition of Sparse data needs a definition of information content of a data array . This raises the first,

Open Question 1:

> What is a useful definition of the information content of an array?

"Useful" means that the information measure can be used to evaluate the performance of various storage, retrieval, and manipulation schemes. Such an information theoretic measure might even suggest some schemes.

In lieu of such a definition, "Sparse Array" is taken to mean either

(1) An array that can be compactly stored due to its very special structure.

Ex: An arithmetic progression.

or

(2) A regular homogeneous array comprised of two classes of data. Class one is comprised of one or a few elements repeated often. Class two is general. The first class covers most of the array, the second class is rare in the array.

Ex: An LP array with K non-zero elements in each row and N-K elements zero. K is less than 5 percent of N. In this case, the only repeated (class one) element is the zero but it covers more than 95 percent of the array.

Ex: A character array with only one long line, and the rest short, such that most of the array is blank. The blank is class one, the non-blanks are class two.

This definition derives directly from Sparse Matrix Research (SMR) [1-12]. However, notice that in APL there is no restriction to two dimensional arrays.

2. RELATED WORK

The SPARCOM paper [13] is an example of the special sparse case of bit arrays. Ashany uses a Gustavson 'Row Pointer Column Index' (RPCI) scheme [10, 11, 12]. It is not yet clear whether any substantial gain over the current implementations can be realized by generalized sparse storage schemes in the bit array case. It is suspected that this special case will require special methods involving special hardware for any major improvement in performance.

In this case the arrays are primarily two dimensional and homogeneous.

3. SPARSE MATRIX RESEARCH (SMR) AND SPARSE APL

Sparse data types in APL have been mentioned in passing [14] but we know of no work on the subject.

Sparse matrix research (SMR) is primarily motivated by the need to solve large systems of algebraic equations in the real domain, where the matrix of coefficients is sparse and real, i.e. contains many zero entries (class one data) and few non-zero reals (class two data). Many areas of commercial data processing exhibit sparse array characteristics such as in material processing and the matrix form of double entry (or double classification) bookkeeping. Practically no work has been done in applying sparse array ideas to these commercial data processing areas. *

The field of sparse matrix research and development is large in terms of the number of investigators and publications available. The Bibliography contains a short list of books and survey papers on the subject [1-12]. This is a sufficient starter for anyone wishing to delve deeper into the subject.

Open Question 2:

> What are the areas in Commercial data processing that would benefit by a Sparse APL?

3.1 PIVOTING

For the purpose of relating SMR to sparsity in APL, recall that the solution of a system of algebraic equations involves scalar multiplication of a row vector and row-row addition or subtraction. Looked at in this way, admittedly over-simplified, SMR is concerned with efficient storage, retrieval, and execution of

    ROWI -- ROWI + K x ROWJ.

----

* We thank Gene McDonnell for pointing out these commercial data processing examples.

This involves a very limited subset of APL capabilities.

3.2 TRANSPOSITION

A little deeper into SMR it is found [12] that the operation of transposition is also of central concern. This concern derives from the bias some storage schemes [6] have for efficient row-oriented storage and retrieval. When column operations are required, it is often best to transpose the array in situ and then to proceed with row retrieval and processing rather than implement column operations directly.

3.3 SPARSE ARRAY I/O

There is other SMR work relevant to Sparse APL, such as work on the problem of input of a sparse matrix, the display of its structure (zero non-zero pattern), and the printing of sparse matrices. See the MPSX system [15] for one set of solutions to these problems. The input of (I, J, element) implies and is implied by the storage scheme employed in MPSX. The PICTURE capability is a character pattern print out generator. This often requires laying the printout sheets on the floor of a conference room and standing on a ladder in order to get a "gestalt" of the structure.

All totaled, our opinion is that the majority of the problems in defining a sparse APL are precedented in SMR but are much more general. For example, Gustavson's work on transpose [12] is important in determining the direction of attack on structural primitives such as transpose, rotate, reversal, and reshape, but there are new difficulties encountered in APL.

3.4 MULTIDIMENSIONAL ARRAYS

Although most of the Sparse matrix literature and techniques were developed for rectangular matrices because the applications generate them, there are applications that generate multidimensional arrays. This work appears in the literature dealing with tensor product formulations in differential equations, for example. We have not yet looked deeply into this application area, but so far we have not found relevant references on sparse implementations. Any such would be appreciated.

4. SUGGESTED RESEARCH DIRECTION

We suggest an investigation into transferring whatever is relevant from SMR to the definition of a sparse array implementation of APL. We are primarily concerned with Sparsity in the second definition form, as given in section 1. above. Our specific suggestion is toward generalizing Gustavson's row pointer column index (RPCI) [10] scheme to multi-dimensional regular homogeneous arrays and then enlarging his transposition scheme [12] to other structural primitives if possible. In addition to this, one must examine the other primitives for efficiency if they were to act on arrays stored in RPCI form. This leads to

One way to generalize RPCI is to map any array onto a matrix maintaining row and column order (see 5.1 below). This leads to

Open Question 3:

> Is generalized RPCI critically sub-optimal for some APL primitive?

4.1 CHOICE OF APL PRIMITIVES FOR STUDY

We suggest using information developed by Saal and Weiss [16] and others on frequency of use of APL primitives. One must choose the most frequently used primitives for analysis. Eventually, all primitives should be analyzed for the effects of this suggested sparse storage scheme.

4.2 FEASIBILITY OF VERIFICATION OF SPARSE ARRAY TECHNIQUES

By writing APL functions that perform the actions of selected APL primitives on data stored in this chosen sparse data storage technology, these functions would verify that the primitives could, in fact, be implemented. The functions would also act as prototypical implementations of the primitives in a new Sparse Array APL interpreter.

It is possible that a large subset of primitives will have to be replaced for an experiment to be a reasonable test and generate meaningful data.

It is also necessary to provide I/O conversion routines to communicate outside APL, including terminal display. This necessitates defining an extension to the actions of the Format primitive, (but not its function in the language) and the system formatting functions.

Open Question 4:

> Is there an efficiency to be gained by explicit user specification that an array should be stored sparsely as opposed to a hidden scan/decision?

5. OTHER RELATED FIELDS

5.1 GENERALIZED ARRAYS AND SPARSE ARRAYS

We have noted a close relation between Generalized Arrays or Array Theory (T. More Jr. [17]) and Sparse arrays. Any Generalized Array can be imbedded in a large enough regular array of sufficient dimension though, in general, it won't be homogeneous. Quite often this imbedding generates a sparse array where the "fill" elements are class one data (zero, blank, null-element).

Conversely, many classical sparse matrices, and many projected sparse APL arrays are of "special structure" and can be specified as compact general arrays. For example, a tri-diagonal matrix can be specified as a triple of different length vectors catenated with a single triple of offsets which determine which three matrix diagonals are being represented.

This has led us to a conjecture, with Jim Brown of the IBM Research Division, that a hybrid pointer scheme combining our suggested modification of RPCI with Brown's pointer structure for generalized arrays might be a very efficient way to store sparse data for APL. See [18].

5.2 EXTENDABLE ARRAY STORAGE

We would expect to find a close relationship between the theory of extendable arrays as it is developing, [19], and the final implementations of Sparse arrays.

Open Question 5:

> What is the relationship between extendable arrays storage techniques and the techniques used for extending the storage areas of sparse arrays as they fill in?

5.3 OTHER LANGUAGES

There is a continuing movement towards including distributed arithmetic into languages other than APL. For example, there is the extension of +,-,/,*, in PL/I to array and structure arguments.

Any results uncovered for these operators in APL would apply also to any language using distributed arithmetic.

5.4 INDEXING

Indexing data is a general Computing Science problem that generates many interesting, and for Sparse APL, possibly relevant results [19, 20, 21].

5.5 DATA COMPACTION

The problem of efficiently storing and fetching Sparse arrays appears in the data communication context, especially where data are distributed over a network and need to be compactly transmitted. There are schemes for coding transmitted data (see any recent text on information theory, especially the sections on Huffmann coding).

We are concerned here with compacting the data before presentation to those schemes. We expect a savings by pre-transmission context dependent compression, which would be allowed by a Sparse data storage technology.

The relationship between context dependent compaction and methods currently in use in data communications is a problem of independent interest in data communications.

Open Question 6:

What is this relationship?

5.6 MACHINE DESIGN

Various Sparse data storage techniques have been developed for use in advanced hardware. Conversely, various proposals for advanced hardware have implied modified storage techniques.

For example, if a bit string defines the placement of the non-zero elements in a vector, the ANDing of bit strings associated with two vectors defines a new bit string that defines which corresponding elements in the two vectors are non-zero.

In an APL inner product primitive or in a hardware device for accumulating products, the resultant bit string can be used to suppress all multiplications except the ones with two non-zero factors.

In both the APL software and the hardware contexts, it is clear that reduced operation count implies improvement in performance if the data storage technology overhead is not too great.

Open Question 7:

What hardware modifications would most cost effectively facilitate Sparse array computation, especially of Sparse bit arrays?

6. SUMMARY

Sparse APL raises many new and interesting questions of both theoretical and practical interest.

The precise specification and then solution of these problems should have wide effects because the problems seem to be generic Computing Science problems.

We suggest broad participation in the specification and solution of these interesting problems.

7. BIBLIOGRAPHY

7.1 BOOKS ON SMR

[1] Sparse Matrix Proceedings IBM RA-1. Ralph A. Willoughby, Ed. IBM Yorktown Heights, N.Y. (1969).

[2] Large Sparse Sets of Linear Equations. J. K. Ried, Ed. Academic Press, N.Y. (1971).

[3] Sparse Matrices and Their Applications. Donald J. Rose and Ralph A. Willoughby, Eds. Plenum Press, N.Y. (1972).

[4] Sparse Matrices, Reginald P. Tewarson. Academic Press, N.Y. (1973).

[5] Sparse Matrix Computations. James R. Bunch and Donald J. Rose, Eds. Academic Press, N.Y. (1976).

7.2 SURVEY REPORTS ON SMR

[6] "A Survey of Indexing Techniques for Sparse Matrices" in ACM Computing Surveys Vol. 5 No. 2, pp 109-133 (June 1973).

[7] A Survey of Sparse Matrix Research. C.S.S. 28 by I. S. Duff. (Feb. 1976). Computer Science and Systems Division Building 8.9 AERE Harwell, Didcot, OXON, England.

[8] Sparse Matrices C.S.S. 31 by J. K. Ried. (May 1976). Computer Science and Systems Division Building 8.9 AERE Harwell, Didcot, OXON, England.

7.3 SOME TECHNICAL PAPERS ON SMR

[9] Some Design Features of a Sparse Matrix Code by I. S. Duff and J. K. Ried C.S.S. 48 (July 1977). Computer Science and Systems Division Building 8.9 AERE Harwell, Didcot, OXON, England.

[10] "Some Basic Techniques For Solving Sparse Systems of Linear Equations" by Fred G. Gustavson, appearing in 3 .

[11] An Efficient Algorithm To Perform Sparse Matrix Multiplication by Fred. G. Gustavson. IBM RC 6176 IBM Yorktown Heights, N.Y. (1976).

[12] Permuting Matrices Stored In Sparse Format by Fred G. Gustavson. RC6181 IBM Yorktown Heights, N.Y. (1976).

7.4 OTHER

[13] SPARCOM by Ron Ashany. TR 00.2774 IBM Poughkeepsie (1976).

[14] "What's Wrong With APL?" by Philip S. Abrams in APL 75 SEAS SHARE Conference, Pisa Italy, June 1975, pp. 1-8. Available from the ACM NYC, NY.

[15] IBM Mathematical Programming System Extended/370 (MPSX/370) Program Reference Manual SH19-1095.

[16] An Empirical Study of APL Programs. by Harry J. Saal and Zvi Weiss in Computer Languages Vol. 2. pp. 47-59. Pergamon Press. Great Britain (1977).

[17] A Theory of Arrays With Application To Databases by T. More Jr. IBM Cambridge Scientific Center G320-2106 (1975).

[18] A Generalization of APL. Ph.D. Thesis by Jim Brown. Syracuse Univ. Syracuse, N.Y. (1971).

[19] Techniques For Storing Extendible Arrays by Arnold L. Rosenberg and Larry J. Stockmeyer. RC 5643 IBM Yorktown Heights, N.Y. (1975).

[20] Methods of Array Indexing by K. L. Doty and V. S. Moore TR 54.069 IBM GSD Boca Raton, Fla. (1976).

[21] An Analysis of Indexing Strategies by George Markowsky. RC 6038 IBM Yorktown Heights, N.Y. (1976).

SCIENTIFIC CENTER REPORT INDEXING INFORMATION

1. AUTHOR(S) : Ronald I. Frank	9. SUBJECT INDEX TERMS APL Sparse Array Storage Mapping 05 - Computer Application 16 - Mathematics 21 - Programming
2. TITLE : Sparsity and APL	
3. ORIGINATING DEPARTMENT: Cambridge Scientific Center	
4. REPORT NUMBER: G320-2125	
5a. NUMBER OF PAGES: 12	5b. NUMBER OF REFERENCES: 21
6a. DATE COMPLETED: 2/78	6b. DATE OF INITIAL PRINTING: 12/78
6c. DATE OF LAST PRINTING:	

7. ABSTRACT :

A definition of Sparse APL is proposed. A review of Sparse Matrix Research is presented and related to the current APL language and systems. Presented next is a discussion of the possible extensions of classic Sparse Matrix methods that would be required in a "Sparse Array" implementation of APL. This includes a discussion of a suggested research direction. Some relationships of Sparse APL to other fields outside of APL are mentioned. Finally, a list of open questions is interposed for anyone interested in pursuing some of these questions.

8. REMARKS :

1977 IBM CAMBRIDGE SCIENTIFIC CENTER OUTSIDE PUBLICATIONS

A. P. DEMPSTER, M. SCHATZOFF & N. WERMUTH, A Simulation Study of Alternatives to Ordinary Least Squares, Journal of the American Statistical Association, March 1977, Vol. 72, No. 357, 77-106

Y. BARD, The Modelling of Some Scheduling Strategies for an Interactive Computer System, in: Computer Performance, North Holland Publishing Co., 1977, 113-137

1978 IBM CAMBRIDGE SCIENTIFIC CENTER OUTSIDE PUBLICATIONS

Y. BARD, M. SCHATZOFF, Statistical Methods in Computer Performance Analysis, in: Current Trends in Programming Methodology, Vol. III, Software Modelling, Prentice-Hall, Inc., 1978, 1-51

Y. BARD, The VM/370 Performance Predictor in Computing Surveys, Vol. 10, No. 3, September 1978 (pp. 333 - 342)

Y. BARD, An Analytic Model of the VM/370 System in IBM Journal of Research and Development, Vol. 22, No. 5, September 1978 (pp. 443 - 586)

IBM® Cambridge Scientific Center, 545 Technology Square, Cambridge, Massachusetts 02139