Overview of Mathematical Symbol Manipulation

Stanly Steinberg
Department of Mathematics and Statistics, University of New Mexico
Albuquerque, New Mexico 87131, 505-277-4733, USA

The currently available software and hardware has made mathematical symbol manipulation a more fruitful and productive activity than just a few years ago. The combination of the new inexpensive hardware and software will certainly cause a substantial growth in the user community. In the present paper we discuss several of these developments.

1. Introduction

The idea of using a computer to manipulate symbols has been around as long as the idea of using computers to manipulate numbers. Currently there are many areas in which computers are used to manipulate symbols rather than numbers; for example, text processing, artificial intelligence and symbolic mathematics. The interest here is in programs that do symbolic mathematics; that is, programs that do many of the nonnumeric calculations from high school algebra, university calculus, ordinary differential equations and many other calculations usually thought of as being within the exclusive domain of humans. The programs that perform such calculations are frequently called symbol manipulators. Symbol manipulators were certainly capable of doing interesting problems in the early 1960's; though used extensively, they are just now gaining general acceptance by the computing public. It appears that improvements in software and hardware will encourage more extensive use of the symbol manipulation technology. For a far more detailed discussion of symbol manipulation and symbol manipulators see the text by Buchberger et al.

Symbol manipulation programs can be conveniently divided into two categories; special purpose and general purpose. The interest here is in general purpose symbol manipulators, but it is important to realize that special purpose programs have played a crucial role in certain scientific areas. Symbol manipulators have made significant contributions to a wide range of problems as a brief look at the conference proceedings listed below will show. The articles by Elvey, Ogilvie, Pavelle et al., Steinberg, and Stoutemeyer provide a more detailed overview of general purpose symbol manipulators than will be given here.

2. The main symbol manipulators

Some of the general purpose symbol manipulators are MACSYMA, Maple, Mathematica, muMATH, REDUCE, SMP and Scratchpad II (addresses are given below). The program REDUCE has been available to the public for many years, muMATH has been available for several, MACSYMA and SMP were made available to the general public in 1983, Maple was released in 1984. Scratchpad II and Mathematica are just now being released. Until 1983 MACSYMA, the most powerful of the symbol manipulators, was available on computers at MIT to which many users around the country had access using the ARPA net. MACSYMA is now available for many different computers; it is being sold (at modest cost to universities) by Symbolics. Maple, SMP, and Mathematica have nearly as many (and in some areas more) facilities as MACSYMA. There are probably more users of REDUCE than of all other symbol manipulators combined. The facilities in REDUCE are modest when compared with those of MACSYMA. On the other hand, the program is well documented, easy to use, and powerful, so if a problem falls within its domain, it is perhaps the best program to use. A user library is being built which will make this program much more useful. Maple requires little in terms of machine resources, but still has excellent facilities. Thus it is excellent for students in a resource-shared environment. The muMATH program is designed to run on small micro-computers and seems not to be powerful enough to do any large scientific computing. This program could be used in high school and college to interest students and faculty in symbol manipulation. High school students are probably prepared to do nontrivial symbolic problems before they can do nontrivial numerical problems. In early 1987, Hewlett-Packard introduced a hand-held calculator (model 28C for approximately two hundred dollars) that is capable of doing nontrivial symbolic calculations.

3. Limitations

It is a common experience for a person who is first using a symbol manipulator to feel that the program is incredibly powerful. In fact these programs are very powerful but they also have their limitations. Symbol manipulation programs are capable of doing infinite precision rational arithmetic, algebraic simplification, expanding and factoring, finding greatest common denominators and other operations of the type found in high school algebra courses. The programs are also programming languages and this adds much to their power. Some of the programs are capable of handling expressions that contain millions of characters and perform, in seconds, calculations that would require humans many tedious hours. The programs also 'know' how to differentiate and integrate. Although it is not well-known, there are algorithms for integrating certain large classes of functions. When an integration problem falls in one of the known classes, the programs are capable of doing integrations where the answer may fill several printed pages and appear impossible to do by hand. It is experience with these types of problems that leads to optimism about symbol manipulation. However, it is also not difficult to find examples that humans can do easily and the existing programs cannot.

As users of symbol manipulation programs become more experienced they soon discover there are many limitations to the symbol manipulators. The programs may run out of memory, may take too much time, may not contain a needed algorithm or may be set up in such a way as to make some problems unreasonably difficult to solve. Not unlike *stability* in numerical calculations, *intermediate expression swell* plays an important role in symbolic calculations. Intermediate expression swell refers to a situation where the statement and results of some calculation are rather small but some intermediate steps in the calculation produce very large expressions. Intermediate expression swell can be reduced by careful analysis and programming of problems (see Steinberg and Roache), but some form of the difficulty is inherent in many interesting problems. Consequently many problems require large amounts of memory and time to be completed.

Perhaps the most important problem with symbol manipulation programs is the limited amount of mathematics that they 'know' (see Steinberg). With certain noteworthy exceptions, the programs know relatively little about junior- and senior-level (college) applied mathematics. The MACSYMA packages for solving ordinary differential equations and tensor manipulations are two of the exceptions. Although there are many symbol manipulation users who are creating programs using advanced level mathematics, there is too little effort being expended on incorporating this mathematics into the main body of the programs. There are libraries of user-contributed programs but this seems to be a rather ineffective way of making a wide range of facilities available to the user community. Some workers have made some inroads into this problem in certain mathematical areas (for instance, see the thesis of M. Wirth). An example of mathematics being set up in a symbol manipulation program in a way that limits the usefulness of the program is given by the dependencies idea used in the MACSYMA differentiation programs. This idea works well in simple problems but causes considerable trouble for problems involving complicated coordinate changes (see Wester and Steinberg, and Golden). These problems and many others are now under serious attack by the symbol manipulation community as a brief perusal of the conference proceedings listed shows.

4. New developments

The most promising news for the symbol manipulation community is that there are several groups working on a new generation of symbol manipulators; these include a system called Scratchpad II at IBM (see Jenks), a one called Newspeak at UC Berkeley (see Fateman, and Williamson), and one called Views at Tektronix (see Abdali, Cherry, and Soiffer). The system will no longer be based on LISP or other standard languages but instead will be based on a new language that will attempt to merge the underlying mathematics and mathematical objects with the algorithms and data types and the hardware to provide a substantial gain in speed and reduce memory requirements over the existing systems. Because much care is being exercised to embody good mathematics in the underlying language, there seems little doubt that many problems that are now tedious or impossible to do with the current systems will be manageable with the new system. Considerable effort is being directed toward developing user interfaces that are appropriate for both the experienced user and the novice user. Another component of the project will bring together experienced users of symbol manipulation systems who have a strong applied mathematics background in order to incorporate higher levels of mathematics in the new system.

Symbol manipulation programs are large and need to be able to handle large expressions. As the size of the expressions grows so does the use of computer time; it is not uncommon to hear experienced users talk about problems that run many hours on a large computer. Thus symbol manipulators place a considerable strain on the computing resources. Because of this demand on computing resources, many symbol manipulators used to run in batch mode or on special machines as MACSYMA did at MIT. Batch mode symbolic computing turns out to be unsatisfactory because of another problem inherent in symbolic computing; the powerful symbol manipulation programs are not predictable enough to be easily programmed without first testing various ideas interactively. All of the programs mentioned above are now interactive.

5. Hardware

An important hardware development for the symbol manipulation community was the appearance of the VAX computer. The ability of these machines to automatically page information on the disk to the main memory allows the machine to easily run large interactive jobs. Maple, REDUCE, SMP and MACSYMA now all run on these machines. If a VAX-class computer has four or more megabytes of main memory and is lightly loaded, then MACSYMA will run very nicely and Maple will run easily. However, if the machine is heavily loaded or has a small main memory, then the performance of MACSYMA and the machine is seriously degraded while small jobs in Maple cause no more problem than an editor.

Another hardware related problem with the current symbol manipulators is their input and output. The input is typed in a linear FORTRAN like format. Programs that can understand written formulas are feasible but expensive and inconvenient. However, once some information is typed into a symbol manipulator and displayed on a graphics screen, then a 'mouse' or light pen can be used to manipulate the information or direct the program to manipulate the information. The output of current programs is usually a relatively primitive two-dimensional format. However, some programs are capable of producing text-book quality output by using a laser printer, and similar results can be obtained by using a high resolution graphics screen. Since human vision plays an important role in understanding formulas, high quality output will greatly enhance the usefulness of symbol manipulation programs. Such technology will allow symbol manipulators to mimic pencil and paper calculations which will make the use of the programs more intuitive. The system being developed by the Fateman group will take advantage of graphics screens and the 'mouse' technology. Some prototype systems were discussed at the Waterloo conference.

An optimal environment for symbol manipulation is a workstation; a machine with 4 megabytes of main memory, large hard disk drive, 16 megabytes of virtual address space, high resolution graphics screen, and a 'mouse' are now available and provide better service for a single user than a large time-shared machine. A network of such machines with a common disk drive would be ideal for a community of users. All of the symbol manipulators mentioned above now run on workstations. Symbolics has MACSYMA running on their LISP machine; a LISP machine is a large workstation and is excellent for doing symbolic calculations.

6. References

Journals

Computer Algebra Systems in Education Newsletter, Dept. of Mathematics, Colby College, Waterville, ME 04961.

Journal of Symbolic Computation, B. Buchberger, Editor, Johannes Kepler University, A4040 Linz, Austria. Published by Academic Press.

MACSYMA Newsletter, Symbolics, Inc., Eleven Cambridge Center, Cambridge, MA 02142.

Scratchpad II Newsletter, Computer Algebra Group, Knowledge Systems, Computing Technology Department, IBM Thomas J. Watson Research Center, Box 218, Yorktown Heights, NY 10598.

SIGSAM Bulletin, Special Interest Group on Symbolic & Algebraic Manipulation of the ACM (Association for Computing Machinery). 11 West 42nd Street, New York, NY 10036.

Papers

S. Kamal Abdali, Guy W. Cherry, and Neil Soiffer, An object oriented approach to algebra system design, *Proceedings of the 1986 symposium on Symbolic and Algebraic Computation*, Association for Computing Machinery, 1986.

R.D. Drinkard, Jr., N.K. Sulunski, MACSYMA: A program for computer algebraic manipulation (demonstrations and analysis). New London Laboratory, Naval Underwater Systems Center, New London, Connecticut 06320.

J.S.N. Elvey, Symbolic Computation and Constructive Mathematics, Research Report, Dept. of Computer Science, Univ. of Waterloo, Waterloo, Ontario, Canada, 1983.

R.J. Fateman, My view of the future of symbolic and algebraic computation, *SIGSAM Bulletin 18*, 1984, 10-11.

Jeffery P. Golden, Differentiation of unknown functions in MACSYMA, *SIGSAM Bulletin 19*, 1985, 19-24.

M.A. Hussain, B. Noble, Applications of MACSYMA to calculations in applied mathematics. General Electric Co.

R.D. JENKS, 11 keys to new scratchpad, Proceedings of EUROSAM 84, edited by J. FITCH, Springer-Verlag, New York, 1984.

J.F. OGILVIE, Applications of computer algebra in physical chemistry, *Computers and Chemistry 6*, 1982, 169-172.

R. PAVELLE, M. ROTHSTEIN, J. FITCH, Computer algebra, *Scientific American*, Dec. 1981.

S. STEINBERG, Mathematics and symbol manipulation, *SIGSAM Bulletin 16*, (1982), 11-15.

S. STEINBERG, P. ROACH, Symbolic manipulation and computational fluid dynamics, *Journal of Computational Physics 57*, (1985), 251-284.

D.R. STOUTEMYER, Symbolic computation comes of age, *SIAM News 12*, 1979.

M. WESTER, S. STEINBERG

An extension to MACSYMA's concept of functional differentiation, *SIGSAM Bulletin 17*, 1983, 25-30.

A survey of symbolic differentiation implementations, 1984 MACSYMA User's conference.

C. WILLIAMSON, JR., Taylor Series solutions of explicit ODE's in a strongly typed algebra system, *SIGSAM Bulletin 18(69)*, 1984, 25-29.

M.C. WIRTH, On the automation of computational physics, Lawrence Livermore Laboratory, 1980.

Texts

A.K. AKRITAS, Text to be published in 1988, Department of Computer Science, University of Kansas.

B. BUCHBERGER, G.E. COLLINS, R. LOOS, R. ALBRECHT (Editors), Computer Algebra, Symbolic and Algebraic Computation, Springer-Verlag, New York, 1982.

B.F. CAVINESS, R.P. GILBERT, R. SHTOKHAMER, *An Introduction to Applied Symbolic Computation Using MACSYMA*, course notes, Department of Computer Science, University of Delaware.

J. COHEN, *Introduction to Computer Symbolic Manipulation, Applications, and Algorithms*, Department of Mathematics and Computer Science, University of Denver.

R.E. ZIPPEL, Algebraic Manipulation (Course notes) Dept. of Computer Science, MIT, 1982.

K.O. GEDDES, Algebraic Algorithms for Symbolic Computation (Course Notes) Dept. of Computer Science, University of Waterloo.

R.H. RAND, Computer Algebra in Applied Mathematics: An introduction to MACSYMA, Pittman, Marshfield, 1984.

Conferences and Proceedings

J. FITCH (Editor), EUROSAM 84, International Symposium on Symbolic and Algebraic Computation, Cambridge, England, July 1984, Lecture Notes in Computer Science, 174, Springer-Verlag, New York, 1984.

V.E. GOLDEN (Editor), Proceedings of the 1984 MACSYMA User's Conference, Schenectady, New York, July, 1984. For copies send $30.00 to M.

McGinn, General Electric Co., Corporate Research and Development Center, P.O. Box 8, Bldg. K-1, Room 3A15, Schenectady, NY, 12301.
B. BUCHBERGER (Editor), EUROCAL '85, Linz, Austria, April, 1985.
S. HITOTUMATU (Chairman), Fifth RIMS Conference on Formula Manipulation and Its Application to Mathematical Study, 1985, Kyoto, Japan.
B. CHAR (Editor), SYMSAC '86, Proceedings of the 1986 Symposium on Symbolic and Algebraic Computing, ACM, 1986.
T. SASAKI (Chairman), Sixth RIMS Conference, 1986, Kyoto, Japan.
W. LASSNER (Chairman), EUROCAL '87, Leipzig, GDR, June 1987.
R. JANSSEN (Editor), Trends in Computer Algebra, Lecture Notes in Computer Science, Springer Verlag, 1987.
A. MIOLA (Chairman), FIJC-88, Roma, Italy.

Symbol Manipulation Programs

MACSYMA: Computer Aided Mathematics Group, Symbolics Inc., 11 Cambridge Center, Cambridge, MA 02142.

DOE MACSYMA: JAN MOCKLER, National Energy Software Center, Argonne National Laboratory, 9700 S. Cass. Ave., Chicago, Illinois 60439. LEO HARTEN, Paradigm Associates, Inc., 29 Putnam Avenue, Suite 6, Cambridge, MA 021329.

Maple: WATCOM Products Inc., 415 Phillip St., Waterloo, Ontario, Canada N2L 3X2.

Mathematica: Wolfram Research, Inc., P.O. Box 6059, Champaign, Illinois 61821.

muMATH: The Soft Warehouse, 3615 Harding Ave., Suite 505, Honolulu, Hawaii 96816.

REDUCE: A.C. HEARN, The Rand Corporation, 1700 Main St., P.O. Box 2138, Santa Monica, CA 90406.

SMP: Inference Corporation, Computer Mathematics Group, Suite 501, 5300 West Century Blvd., Los Angeles, CA 90045.

Scratchpad II: R. JENKS, Computer Algebra Group, IBM T.J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598.