COMPUTER PHYSICS COMMUNICATIONS 3, SUPPL. (1972) 75–78. NORTH-HOLLAND PUBLISHING COMPANY

ALGEBRAIC TECHNIQUES

M. VELTMAN

Institute for Theoretical Physics, University of Utrecht, The Netherlands

In the past 15 years computer programs have been developed that are capable of performing algebraic manipulations. One may ask what impact this has had on physics, and to what extent one could have done without. In this talk I will try to give an impression concerning the achievements in this field.

If an algebraic expression must be worked out according to well defined rules one can in principle have this done by computer. To this purpose one must have a program that can understand expressions, and offers possibilities to define the work to be done on those expressions.

Many programs doing this sort of work have been written [1]. In fact, there are probably more programs than problems. And it is a sad thing that most of these programs cannot solve any of the problems.

Broadly speaking programs may be used

- for fun,
- for educational purposes,
- to check calculations done by hand,
- to solve problems that are in one sense or another too big for human evaluation.

In this talk the emphasis will be on the latter use, as in that case the eventual impact on physics would be largest. With this limitation in mind very few programs remain of interest, depending on where one draws the line of what is possible for a human being. Also it is very hard to estimate the factor of human ingenuity, that may render "impossible" problems possible.

To illustrate this I would like to consider a specific example that has been advertised as an example of algebraic computation. It is a problem in celestial mechanics, namely the computation of the so-called f and g series [2]. Mathematically the problem presents itself as an iteration procedure:

$$f_0 = 1 \;; \qquad g_0 = 0 \;;$$

$$f_i = \dot{f}_{i-1} - \mu g_{i-1} \;; \qquad g_i = f_{i-1} + \dot{g}_{i-1} \;.$$

The f_i and g_i depend on certain variables μ, σ and ϵ, and one has:

$$\dot{\mu} = -3\mu\sigma \;; \qquad \dot{\sigma} = \epsilon - 2\sigma^2 \;; \qquad \dot{\epsilon} = -\sigma(\mu + 2\epsilon) \;.$$

The calculation is entirely trivial, but becomes cumbersome after 10 or more iterations. The result after 5 iterations is for example:

$$f_5 = -45\,\sigma\epsilon\mu - 15\,\sigma\mu^2 + 105\,\sigma^3\mu \;;$$

$$g_5 = -45\,\sigma^2\mu + 9\,\epsilon\mu + \mu^2 \;.$$

A rough estimate yields that a man would need about 35 hours to do 20 iterations (handling approximately 3000 terms), while the CDC 6600 needs about 8.5 seconds, doing mostly input–output. Thus, if we wanted to boast to an outsider we could say that the computer works 10000 times faster than a man.

However, let us now assume that this man thinks [3] a little bit (say 10 minutes). He will straightaway invent another method. Since the f and g are polynomials in μ, σ and ϵ one may write:

$$f_i = \sum_{l,m,n} a_{ilmn}\mu^l\sigma^m\epsilon^n \;, \qquad g_i = \sum_{l,m,n} b_{ilmn}\mu^l\sigma^m\epsilon^n \;,$$

where the a and b are numerical coefficients. Next inserting these equations in the recursion equation one obtains recursion equations for the numerical coefficients a and b:

$$a_{i+1,l,m,n} = -(3l + 2m + 2n)a_{i,l,m-1,n} + m\,a_{i,l,m+1,n-1} - n\,a_{i,l-1,m-1,n+1} - b_{i,l-1,m,n} \;.$$

$$b_{i+1,l,m,n} = a_{i,l,m,n} - (3l + 2m + 2n)\, b_{i,l,m-1,n}$$
$$+ m\, b_{i,l,m+1,n-1} - n\, b_{i,l-1,m-1,n+1} \,.$$

Starting with $a_{0000} = 1$, all other a's and b's being zero the problem becomes a very simple numerical problem. In fact, everything can be computed by a trivial Fortran program in a fraction of the time needed for algebraic evaluation. And our man needs to know only Fortran and can use the computer of the grocery around the corner. Which shows that this problem does not need an algebraic program at all.

There is another factor that becomes evident the very moment that one starts doing algebra on the computer, namely the circumstance that results become very voluminous, much more so than if done by hand. This is because it is extremely difficult to program the computer to try to write results as easy as possible. In fact what is involved is pattern recognition. For instance the computer should be able to recognize that $a^2 + 2ab + b^2 = (a+b)^2$. Of course this particular one is easy, but if you have a large number of terms, with all kind of additional factors, and where one does not know beforehand what regularities appear then it becomes an almost impossible task with present day hardware. The consequence is that for problems that exceed human possibilities the computer generally needs a very large amount of storage space. Systems like LISP, that are very uneconomical in storage are here at a severe disadvantage. A rough estimate yields that for storing the same expression LISP needs about 20 times as much storage as a machine language based system. Due to this kind of complication algebraic programs written in some so-called higher level language (rather than directly in machine language) appear to be barely capable of attacking problems exceeding human possibilities. As a case in point I can quote some work done in general relativity [4]. There one meets the problem of deriving the Ricci tensor from the metric tensor. Workers in this field have developed LISP based programs, but in all but the simplest cases they get stuck on storage and CP time problems. Then one must use symmetry properties of the various objects as much as possible, and on top of it fiddle with the LISP system in order to obtain more efficient storage. An account of such work [5] done on an IBM 360-75 computer quotes 400 seconds for the computation of one component of the Ricci tensor starting from a $g_{\mu\nu}$ with 10 non-zero components. The hand coded program SCHOONSCHIP needed 12 seconds of CDC 6600 CP time per component if computed straightaway (which was possible because no storage complication appeared); exploiting symmetry to some extent reduced this to about one second. It is anybody's guess what would happen if one tries to attack what I call a really big problem * with a system based on a higher level language. It is far from me to belittle the work in general relativity: it appears that the above mentioned achievements are of significance and constitute real progress. It can be mentioned that this work is of the type (a) to be discussed below. Perhaps it could have been done by other methods, or even by hand; it just seems to be on the limit of human possibilities. All this is to lend support to the statement that algebraic programs based on a higher level language will contribute only in a very limited way, and at the cost of considerable extra effort by the user, to the solution of superhuman problems.

Finally we take a look at the application of algebraic techniques in elementary particle physics. The first mention in the literature of algebraic work done by a computer is in a lecture by Feynman at the Conference on relativistic theories of gravitation, Jablonna, Poland, 1962 (quoted in ref. [6], p. 708). This work was done by J. Matthews probably with a program written in BALGOL for the Burroughs 220 in 1960, but no further details are known to me. Since the talk by Feynman on the quantum theory of gravitation has had a very definite impact on elementary particle physics, it seems reasonable to attribute at least some of the credit to the computer, or rather to Matthews and the computer.

Further there are in the field three programs of interest, namely ASHMEDAI written by M. Levine [7], REDUCE written by A. Hearn [8] and SCHOONSCHIP written by this speaker [9]. The main characteristics are:

ASHMEDAI is written partly in FORTRAN, partly in CDC 3000 machine language. It is constructed with a very particular problem in quantum electrodynamics in mind. Using it amounts to

* Using for instance 10 minutes of 6600 time and a considerable amount of disk space in the case of a hand coded program.

writing a Fortran program with calls to many subroutines that perform various tasks.
REDUCE uses LISP as an intermediate language. It is quite language oriented, computer scientists love it.
SCHOONSCHIP is written in CDC 6000, 7000 machine code. It has been constructed with an emphasis on speed and capability for handling very large expressions, while using less than 30 k of core memory. As to the language, it compares to REDUCE in much the same way as FORTRAN compares to ALGOL.

As to Levine's program one is inclined to believe that its use is limited to the author. Here I may be wrong, but that is my impression. Since however it has been exploited to do one of the most spectacular calculations in quantum electrodynamics it must be mentioned because it shows how special purpose programs play a very important role also in the field of algebraic manipulation. It is perhaps interesting to note that both Levine's and my own start in this field originated in a discussion in the sun at the CERN terasse back in the summer of 1963. By the end of '63 he had things running at the IBM 7090 computer at CERN while I did the first problem in the field theory of vector mesons at the IBM 7090 computer at Stanford University with the IBM 7090 version of SCHOONSCHIP. At SLAC I discussed often with Hearn, who at that time, independently, started work on REDUCE. I am sure that this mutual and very pleasant exchange of ideas has led to the fact that in first concept the two programs are quite similar. Of course I think that he is wrong using LISP, but that is another matter.

As to applications by other users I must confess that I am quite vague on this matter. Generally speaking REDUCE has been used by physicists at Stanford and other places equipped with IBM machinery, and also scientists in other fields appear to use it. SCHOONSCHIP in both the IBM 7090 and CDC version has been used heavily at CERN, in Saclay and in Brookhaven.

Work done with these programs can be divided up in various categories:

(a) Calculations of a purely algebraic nature, where one is interested in the structure of the final formulae rather than in the value of the result.
(b) Calculation of certain amplitudes concerning high energy reactions. Such calculations involve manipulation of the Dirac γ-matrices and simple substitutions, but otherwise little analytical work.
(c) Calculation of certain γ-matrix expressions in q.e.d. (quantum electrodynamics) followed up by numerical evaluation of the remaining integrals.
(d) Complete analytical evaluation of diagrams in q.e.d.

As said before, work of type (a) has been done in general relativity, and also in the quantum theory of gravitation. I have made use of the computer in this sense in studying the theory of Yang Mills fields.

A large amount of work of the types (b) and (c) has resulted in a considerable number of publications, by far too many to review them in a systematic way. The Brookhaven Stonybrook group in particular has been quite prolific in domain (b). Some recent publications are given in [10]. Also very interesting is the work of type (c) done by groups at SLAC and at CERN. This is especially interesting because here human ingenuity, algebraic programs and numerical programs were much involved. All three were vital in arriving at the final results. Since excellent reviews have been written on the subject [11] it will not be discussed further here; still I would like to emphasize that the largest and most complicated work is that done by Levine and Wright [12]. A very nice example of how combined analytic and numerical work led to correct equations is the work of Peterman [13].

Work of the type (d) has been performed by Peterman [14] and also by Mignaco and Remiddi [15]. In particular the recent evaluation of the complete fourth order discontinuity by Barbieri et al. [16] is in more than one sense a crowning achievement. Remiddi will present this at this conference. Here I must add that SCHOONSCHIP has profited immensely from the use by Remiddi; it is really due to his suggestions that this program is becoming a powerful tool.

In conclusion it may be said that by now algebraic programs are part of physics, and will certainly become a more and more important tool as more and more scientists get familiar with these techniques. One must not have the illusion that eventually these programs become so easy to use that really everybody can do work with them; the difficulty is much more in the work that has to be done than in the actual formulation of the problem. For this reason

I do not care very much for many of the programs developed without motivation originating in research not directly related to computer science. Even more extreme, it is my opinion that in the end higher level languages such as ALGOL 68 and LISP do more harm than good. They cripple the computer. These things appear to me as pencils with an engine attached to them to move the pencil. Also the terrible importance of language and notation is unclear to me. For example, it leaves me indifferent whether one writes an exponent with ** or raised, as in A^2. This type of progress is strictly for the birds.

I am indebted to Mrs. D. Maison and J. Thornley for stimulating discussions.

References

[1] Papers from the Second Symposium on Symbolic and Algebraic Manipulation, Commun. ACM 14 (1971) 509.
[2] P. Sconzo, A. Le Shack and R. Tobey, Astron. J. 70 (1965) 269.
[3] T. Watson, IBM corporation, unpublished.
[4] R.A. D'Inverno and R.A. Russel-Clark, J. Math. Phys. 12 (1971) 1258.
[5] M.J. Cohen, Thesis (1972), Dept. of Theor. Physics, University of Stockholm.
[6] R.P. Feynman, Acta Phys. Polon. 24 (1963) 697.
[7] M. Levine, J. Comput. Phys. 1 (1967) 454.
[8] A.C. Hearn, Proc. 2nd Symp. on Symbolic and Algebraic manipulation, Commun. ACM 14 (1971) 8.
[9] M. Veltman, CERN preprint (1967).
[10] R.W. Brown and I.J. Muzinich, Phys. Rev. D4 (1971) 1496;
K.O. Mikaelion and J. Smith, Phys. Rev. D4 (1971) 785;
R.W. Brown, A.K. Mann and J. Smith, Phys. Rev. Letters 25 (1970) 257.
[11] B.E. Lautrup, A. Peterman and E. de Rafael, Phys. Rept. 3 (1972) 193.
[12] M. Levine and J. Wright, Phys. Rev. Letters 26 (1971) 1351.
[13] A. Peterman, Phys. Letters 35B (1971) 325.
[14] A. Peterman, the SINAC program, CERN preprint.
[15] J. Mignaco and E. Remiddi, Nuovo Cimento 60A (1969) 519.
[16] R. Barbieri, J. Mignaco and E. Remiddi, to be published.