Algebraic Approximations

David A. McAllester

Artificial Intelligence Laboratory
Massachusetts Institute of Technology

I. INTRODUCTION

There are several artificial intelligence domains, such as electronic circuit analysis or chemical engineering, where handling symbolic mathematics is important. While symbolic mathematics packages exist [I] they often prove inadequate to handle the complexity of the symbolic expressions in such domains [2]. This paper proposes some techniques for using assumptions about relative magnitudes to simplifyexpressions while maintaining a known degree of accuracy. There is a strong analogy between the proposed technique and the use of exponential notation in the approximate representation of real numbers.

The necessity of making assumptions about the relative sizes of the values of variables introduces the possibility that they will be proven wrong during the course of solving some set of equations. Thus a truth maintenance system [3] [4] may prove useful in keeping track of the assumptions underlying specific simplifications and solutions. The number of assumptions needed can be reduced by employing constraint propogation (5) as a primary technique for solving equations.

II. ALGEBRAIC SIMPLIFICATION

The first step in defining an "n significant figure representaion" is to define a canonical polynomial as either an integer constant, a variable, or a polynomial $a_0x^n + a_1x^{n-1} + \dots + a$ where a_0 is not 0 and none of the coefficints a contain the variable x. Notice that every canonical polynomial other than 0 takes on a non-zero value for some (in fact most) interpretations of its variables.

Some way of discussing the relative sizes of variables is needed. Each variable x is associated with an order o(x) which is some power of ten. It Is assumed that x takes on values near o(x). The order function o can be extended to all non-zero canonical polynomials p in such a way that the magnitudes of the values p takes on are usually near o(p). The order o(p) of a non-zero canonical polynomial p is defined to be o(x) for a variable x, 1 for an integer constant, and the maximum of the orders of the non-zero terms for a polynomial of the above form, where the order of a non-zero term a_1x^{n-1} is defined to be $o(a.)o(x)^{n-1}$. A canonical polynomial is called minimal if it is a variable, an integer constant, or if all of its non-zero terms are of the same order and each coefficient of each term is minimal. A minimal polynomial can be thought of as containing only that information relevant to determining the value of the polynomial to within ten percent.

An n significant figure polynomial expansion is a pair $((p_0, p., \dots p\ ) e)$ of an n-tuple of minimal canonical polynomials and an integer e such that for each p_i either p_i equals 0 or $o(p_i)$ equals 10 . Such an expansion Is considered equivalent to the expression $p_0 + p, + \dots + p$ • There is a strong analogy between n significant figure polynomial expansions and n significant figure exponential notaion representations of real numbers. Finally an n significant figure representaion is defined to be a ratio of two n significant figure polynomial expansions and therefore has the following form:

$$\frac{(p_0, p_1, \dots p_n)\ e_1)}{(q_0, q_1, \dots q_n)\ e_2)}$$

The truncation of an m significant figure representaion to an n significant figure representation is defined in the obvious way as the result of dropping m-n low order terms form the numerator and denominator. An n significant figure representation is called an n significant figure approximation of an arbitrary rational expression S if it is a truncation of an m significant figure representation which is exactly equivalent to S.

An expression containing real numbers can be approximately evaluated using procedures which add, subtract, multiply and divide n significant figure approximations of reals. The approximate simplification of rational algebraic expressions can be done in the same way using procedures which add, subtract, multiply and divide the n significant figure representations defined above. It is relatively straightforward to define procedures which compute a canonical polynomial equivalent to the sum, difference, or product of two canonical polynomials. From these one can construct procedures which compute n significant figure approximations to the sum, difference, product or quotient of two n significant figure representations. As opposed to standard numeric operations these symbolic simplification procedures never perform carries.

To fully define an n significant figure simplification procedure it remains only to specify n significant figure representations of numeric constants and variables. A variable x can be represented as the ratio:

$$\frac{((x\ 0\ 0\ \ldots)\ \log_{10}(o(x)))}{((1\ 0\ 0\ \ldots)\ \ 0)}$$

Finally any real number can be approximately represented as the ratio of $((k_0, k_1, \ldots k_n)\ e)$ to 1, where each k_i is some small integral multiple of an appropriate power of ten.

As an example consider the 2 significant figure simplification of the expression:

$$\frac{1}{1/r_1 + 1/r_2}$$

This is the expression for theequivalent resistance of two resistors in parallel. Assume that the order of r_1 is 10^4 while the order of r_2 is 10^2 Thus r_2 is much smaller than r_1 and the equivalent resistance of these two in parallel should be roughly r_2. The following series of expressions shows some of the intermediate states of a simplification of this expression.

$$\frac{\frac{((1\ 0)\ 0)}{((1\ 0)\ 0)}}{\frac{((1\ 0)\ 0)}{((r_1\ 0)\ 4)} + \frac{((1\ 0)\ 0)}{((r_2\ 0)\ 2)}}$$

$$\frac{\frac{((1\ 0)\ 0)}{((1\ 0)\ 0)}}{\frac{((r_1\ 0)\ 4)}{((r_1 r_2\ 0)\ 6)}}$$

$$\frac{((r_1 r_2\ 0)\ 6)}{((r_1\ 0)\ 4)}$$

The final form of this expression indicates the need for some mechanism to remove common factors of the numerator and denominator. Such procedures exist [6] and can be readily incorporated into this framework.

One important application of approximate simplification techniques is the symbolic analysis of complex electronic circuits. Any circuit composed of linear elements can be described by a matix equation Ge = i where G is an admittance matrix, e is a vector of node potentials, and i is a vector of currents. It is often impracticle to compute the exact symbolic solution to such an equation because of the complexity of the exact symbolic inverse of G. It is hoped that by using approximate symbolic arithmetic operations the symbolic computation of matrix inverses will become practical.

III. EQUATIONS, ASSUMPTIONS AND TRUTH MAINTENANCE

Often one is interested in solutions to complex non-linear equations. For example consider the equation:

$$x^3 - x^2 + ax - a^3 = 0$$

Suppose that a is very small, about JO . If x is assumed to be on the order of 1 then by dropping unimportant terms form the equation it can be shown that x is almost exactly 1. Similarly if x is assumed to be about 10 then it can be shown that x is almost exactly a, while for x on the order of 10 x is almost exactly a . Such techniques, though well known in engineering and applied mathematics [7], have not been incorporated into computer symbolic mathematics packages.

In general an expression which is constrained to be zero can be simplified to an n significant figure approximation. To solve for a variable x the the expression should be put in the form ax + b where a and b are n significant figure polynomial expansions. Then x can be set equal to the n significant figuresimplification *of* -b/a. if the original expression can't be simplified to the desired form it may still be possible to simplify it to a product expressions each of which is linear in x. In this case one factor must be zero and each factor can be used to give a possible value for x. Only those solutions which are of a size consistent with the assumed size of x are viable and if there are no such solutions then the assumption about the size of x must be retracted and some other size tried (perhaps the size of one of the solutions for x).

This technique can be extended to systems of equations by employing Gaussian elimination. First an assumed order o(x) is associated with each variable x. Then the equations are solved one at a time, each time for a variable which has not yet been solved for. it is possible that during this process some subset of assumptions about the orders of variables is shown to be inconsistent. When this happens an assumption must be retracted and the simplifications and solutions which depended on that assumption must be redone. A truth maintenance system [3] [A] couid prove useful in keeping track of which simplifications and solutions depend on which assumptions.

Simplifications would be less sensitive to changes in the assumed size of variables if they could be based on assumptions of the form "the order of x is less than the order of y" rather than assumptions about the specific sizes of varibles. A truth maintenance system might again prove useful in keeping track of which simplifications remain valid when assumptions change.

Constraint propogation is a technique for solving sets of equations [5], In constraint propogation some subset of variables are a-priori chosen to be "independent" while others are chosen to be "dependent". Dependent variables are eliminated when they can be expressed directly in terms of independent variables via some exact manipulation of a single equation. Once an exact value for a dependent variable is found it can be simplified to an n significant figure approximation (the simplification of exact values can be very expensive without the use of approximations). This has the advantage that assumptions about the sizes of the dependent variables do not need to be made.

It has the disadvantage that not every set of equations can be solved in this way. It would appear that some mixture of constraint propogatlon and other techniques would be optimal for solving sets of equations under order of magnitude assumptions about variable sizes.

REFERENCES

[1] Mathlab Croup MACSYMA Reference Manual, Version Nine. MIT Laboratory for Computer Science, December, 197 7.

{2] de Kleer, J., Sussman, C. J., "Propogation of Constraints Applied to Circuit Synthesis." MIT AI Lab Memo 485, Cambridge, Mass., 19 78.

I 3] Doyle J. "Truth Maintenance Systems Applied to Problem Solving." MIT AI Lab Technical Report 419, Cambridge, Mass., 1978.

[4] McAllester, D. A. "An Outlook on Truth Maintenance." MIT AI Lab Memo 552, Cambridge, Mass., 1980.

[5] Sussman, G. J., Steele, G. L. Jr., "CONSTRAINTS: A Language for expressing Almost Hierarchical Descriptions." Artif icial Intelligence 14 (1980) 1-39.

[6] Knuth, D. E., The Art of Computer Programming Vol. 2/ Seminumerical Algorithms. Addison Wesley, 1969, 360-37 7".

[7] Bender, C. M., Orszag, S. A. Advanced Mathmatical Methods for Scient ists and Engineers. McGraw-Hill ~ 1978.'