Network Working Group
Request for Comments: 1003

Alan Katz
USC/ISI

March 1987

Issues in Defining an Equations Representation Standard

Status of This Memo

This memo is intended to identify and explore issues in defining a standard for the exchange of mathematical equations. No attempt is made at a complete definition and more questions are asked than are answered. Questions about the user interface are only addressed to the extent that they affect interchange issues. Comments are welcome. Distribution of this memo is unlimited.

I. Introduction

Since the early days of the Arpanet, electronic mail has been in wide use and many regard it as an essential tool. Numerous mailing lists and newsgroups have sprung up over the years, allowing large numbers of people all over the world to participate remotely in discussions on a variety of topics. More recently, multimedia mail systems have been developed which allow users to not only send and receive text messages, but also those containing voice, bitmaps, graphics, and other electronic media.

Most of us in the Internet community take electronic mail for granted, but for the rest of the world, it is a brand new capability. Many are not convinced that electronic mail will be useful for them and may also feel it is just an infinite time sink (as we all know, this is actually true). In particular, most scientists (apart from computer scientists) do not yet use, or are just beginning to use, electronic mail.

The current NSF supercomputer initiative may change this. Its primary purpose is to provide remote supercomputer access to a much greater number of scientists across the country. However, doing this will involve the interconnection of many university-wide networks to NSF supercomputer sites and therefore to the NSF backbone network. Thus, in the very near future we will have a large number of scientists in the country suddenly able to communicate via electronic mail.

Generally, text-only mail has sufficed up until now. One can dream of the day (not so far in the future) when everyone will have bitmapped display workstations with multimedia mail systems, but we can get by without it for now. I believe, however, that the new NSF user community will find one other capability almost essential in making electronic mail useful to them, and that is the ability to include equations in messages.

A glance through any scientific journal will demonstrate the importance of equations in scientific communication. Indeed, papers in some fields seem to contain more mathematics than English. It is hard to imagine that when people in these fields are connected into an electronic mail community they will be satisfied with a mail system which doesn’t allow equations. Indeed, with the advent of the NSF’s Experimental Research in Electronic Submission (EXPRESS) project, scientists will begin submitting manuscripts and project proposals directly through electronic mail and the ability to handle equations will be essential.

Currently, there exists no standard for the representation of equations. In fact, there is not even agreement on what it is that ought to be represented. Users of particular equation systems (such as LaTex or EQN) sometimes advocate just including source files of that system in messages, but this may not be a good long-term solution. With the new NSF community coming on line in the near future, I feel the time is now right to try to define a standard which will meet the present and future needs of the user community.

Such a standard should allow the interchange of equations via electronic mail as well as be compatible with as many existing systems as possible. It should be as general as possible, but still efficiently represent those aspects of equations which are most commonly used. One point to be kept in mind is that most equations typesetting is currently being done by secretaries and professional typesetters who do not know what the equations mean, only what they look like. Although this is mainly a user interface consideration, any proposed standard must not require the user to understand an equation in order to type it in. We are not interested here in representing mathematics, only displayed equations.

In this memo, I will try to raise issues that will need to be considered in defining such a standard and to get a handle on what it is that needs to be represented. Hopefully, this will form the basis of a discussion leading eventually to a definition. Before examining what it is that could be or should be represented in the standard, we will first review the characteristics of some existing systems.

2. Existing Systems

There currently exist many incompatible systems which can handle equations to a certain extent. Most of these are extensions to text formatting systems to allow the inclusion of equations. As such, general representation and standards considerations were not a major concern when these systems were initially designed. We will examine the three main types of systems: Directive systems, Symbolic Language systems, and Full Display systems.

Some text editing facilities simply allow an expanded font set which includes those symbols typically used in mathematics. I do not consider these systems as truly able to handle equations since much of mathematics cannot be represented. It takes more than the Greek alphabet and an integral and square root symbol to make an equations system.

Directive systems are those which represent equations and formating information in terms of directives embedded in the text. LaTex and EQN are two examples. LaTex is a more friendly version of Knuth's Tex system, while EQN is a preprocessor for Troff, a document preparation system available under Unix.

With these Directive systems, it is usually necessary to actually print out the document to see what the equations and formatted text will look like, although there are on-screen previewers which run on workstations such as the Sun. Directive systems have the advantage that the source files are just text and can be edited with standard text editors (such as Emacs) and transferred as text in standard electronic messages (a big advantage considering existing mail interconnectivity of the various user communities). Also, it is relatively easy to make global changes with the help of your favorite text editor (for example, to change all Greek letter alpha's to beta's or all integrals to summation signs in a document. This is generally impossible with the other types of systems described below).

The primary disadvantage of these systems is that writing an equation corresponds to writing a portion of a computer program. The equations are sometimes hard to read, generally hard to edit, and one may make syntax errors which are hard to identify. Also, people who are not used to programming, and typesetters who do not actually know what an equation means, only what it should look like, find specifying an equation in this language very difficult and may not be willing to put up with it.

Full Display Systems are those such as Xerox STAR and VIEWPOINT. The user enters an equation using the keyboard and sees exactly that equation displayed as it is typed. At all times, what is displayed is exactly how things will look when it is printed out. Unfortunately, VIEWPOINT does not allow the user to place any symbol anywhere on the page. There are many things (such as putting dots on indices) which are not possible. For those things which are implemented, it works rather nicely.

Hockney's Egg is a display system which was developed at the UCLA Physics Department and runs on the IBM PC. It has the advantage of being able to put any character of any font anywhere on the screen, thus allowing not only equations, but things like chemical diagrams.

Interleaf's Workstation Publishing Software system is not strictly speaking an equations system, but equations may be entered via a cut and paste method. At all times, what one sees is what will be printed out and one may put any symbol anywhere on the page. The problem with this system is that one HAS TO put everything in a certain place. It sometimes takes an enormous amount of work to get things to be positioned correctly and to look nice.

Generally, Full Display Systems are specific to a particular piece of hardware and the internal representation of the equations is not only hidden from the user, but is in many cases proprietary.

Symbolic Language systems, such as Macsyma and Reduce, also allow the entry of equations. These are in the form of program function calls. These are systems that actually know some mathematics. One can only enter the particular type of mathematics that the system knows.

We next will look at what should be represented in an equations system. We will want a representation standard general enough to allow (almost) anything which comes up to be represented, but does not require vast amounts of storage.

3. What Could be Represented?

We will first examine what it is that could be represented. At the most primative level, one could simply store a bitmap of each printed equation (expensive in terms of storage). At the other end of the spectrum, one could represent the actual mathematical information that the equation itself represents (as in the input to Macsyma). In between, one could represent the mathematical symbols and where they are, or represent a standard set of mathematical notation, as in EQN.

It is useful to think of an analogy with printed text. Suppose we have text printed in a certain font. How could it be represented? Well, we could store a bitmap of the printed text, store characters and fonts, store words, or at the most abstract, we could store the meaning behind the words.

What we actually do, of course, is store characters (in ordinary text) and sometimes fonts (in text intended to be printed). We do not attempt to represent the meaning of words, or even represent the notion of a word. We generally only have characters, separated by spaces or carriage returns (which are also characters). Even when we specify fonts, if a slightly different one happened to be printed out it would not matter greatly.

Equations may be considered an extension of ordinary text, together with particular fonts. However, the choice of font may be extremely important. If the wrong font happens to be printed out, the meaning of the equation may be completely changed. There are also items, such as growing parentheses, fractions, and matrices, which are particular to equations.

We are not interested in representing the meaning of an equation, even if we knew how to in general, but in representing a picture of the equation. Thus, we will not further consider the types of representations made in the Symbolic Language systems. We still have Directive systems and the Full Display systems. We shall assume that both of these will continue to exist and that the defined standard should be able to deal with existing systems of either type.

Assuming we do not want to just store a bitmap of the equation (which would not allow any easy editing or interfacing with existing systems), we are now left with the following possibilities:

1. Store characters, fonts and positions only. Allow anything to be anywhere (this is what Interleaf does).

2. Store characters, fonts, and positions, but only allow discrete positions. This makes it easier to place subscripts and superscripts correctly (this is what Hockney's Egg does).

3. Use a language similar to EQN or LaTex, which has ideas such as subscripts, superscripts, fractions, and growing parentheses. Generally positioning is done automatically when the typesetting occurs, but it is possible to do a sort of relative positioning of symbols with some work.

4. Use a language such as Troff or Tex, which is what EQN and Latex is translated into.

5. Some combination of the above.

In the next section, I will argue for a particular combination of the above as a tentative choice. It may turn out, with more information and experience, that this choice should be modified.

4. What I Think Should be Represented

Let us now take a stab at what sort of standard we should have. First of all, we would like our standard if at all possible to be compatible with all of the existing systems described previously. If the standard becomes widely accepted, it should be general enough not to constrain severely the design of new user interfaces. Thus, while we should provide for efficiently representing those aspects of equations which are commonly used (subscripts, parentheses, etc.) we would like extensions to be possible which enable the representation of any symbol anywhere.

We would like standard mathematical symbols, as well as all Greek and Latin letters to be available. We would also like any required typesetting knowledge to be in programs and not required of the user.

I feel that the exact position of a subscript or superscript should not have to be specified by the user or be represented (unless the user specifically wants it to be). It is nice to be able to place any symbol anywhere (and indeed the standard ought to allow for this), but having to do this for everything is not good. The standard should be able to represent the idea of a subscript, superscript, or growing fraction with no more specification.

My suggestion, therefore, is for something like EQN, but with extensions to allow positioning of symbols in some kind of absolute coordinates as well as relative positioning (EQN does allow some positioning relative to where the next symbol would normally go). This has the advantage that the representation is in ordinary text, which can be sent in messages, the Directive systems can map almost directly into it, and it should allow representation for Full Display systems. The ideas of subscript and superscripts (without having to specify a position), growing parentheses, fractions, and matrices, and special fonts are already there.

Most equations can be specified very compactly within EQN, and if positioning is provided as an extension, exceptions can be handled. (The same could be said for LaTex, however, I consider the syntax there to be somewhat unreadable and prefer EQN. Essentially, either will do).

User interfaces should be able to be easily constructed which would allow one to type in an EQN style specification and have the equation appear immediately on the screen. For non-specialists, it may be better to use existing Full Display systems which are then translated in this EQN like standard (perhaps using a lot of the absolute positioning facility).

5. Conclusions

In summary:

1. A standard for the efficient representation of mathematical equations should be defined as soon as possible in order to allow the interchange of equations in documents and mail messages and the transfer of equations between various existing internal representations.

2. Most equations entry is currently done by people who do not know what the equations mean, and are not programmers. It may be that the optimal user interface for these people is

different than for those who do know mathematics and/or are programmers. An equations standard should not preclude this.

3. The standard should easily handle those aspects of equations which are common, such as the set of things provided in EQN.

4. It should also be possible, however, to place any defined symbol anywhere and the standard should allow this type of specification when needed.

5. As many of the existing systems (all of them if possible) should be able to be translated into the standard.

6. The standard should not make requirements on the user interface such that the user must have much typesetting knowledge. This knowledge should be in the user interface or printing routines.

7. Full Display systems may be best for non-specialists and for non-programmers. Directive systems, perhaps with the ability to preview the final equation on one's screen, may be best for the rest.

8. A distinction should be made between the representation of an equation (which we are dealing with here) and the mathematical knowledge it represents.

I suggest something like EQN as a standard with extensions to allow positioning of symbols in some kind of absolute coordinates as well as relative positioning. This has the advantage that the representation is in ordinary text, which can be sent in messages, the Directive systems can map almost directly into it, and it should allow representation for Full Display systems. The ideas of subscript and superscripts (without having to specify a position), growing parentheses, fractions, and matrices, and special fonts are already there.