More Versatile Scientific Documents
Over-Extended Abstract, working paper

Richard J. Fateman
Computer Science Division, EECS Department
University of California
Berkeley, CA 94720-1776, USA

August 19, 1997

Abstract

The electronic representation of scientific documents: journals, technical reports, program documentation, laboratory notebooks, etc. present challenges in several distinct communities. We see five distinct groups concerned with electronic versions of scientific documents:

- Publishers of journals, texts, reference works, and their authors.
- Software publishers for OCR/ document analysis, document formatting.
- Software publishers whose products access "contents semantics" from documents, including library keyword search programs, natural language search programs, data-base systems, visual presentation systems, mathematical computation systems, etc.
- Institutions maintaining access to electronic libraries, which must be broadly construed to include data and programs of all sorts.
- Individuals and their programs acting as their agents who need to use these libraries, to identify, locate, and retrieve relevant documents.

We need to have a convergence in design and standards for encoding new or pre-existing (typically paper-based) documents in order to most efficiently meet the needs of all these groups. Various efforts, some loosely coordinated, but just as often competing, are trying to set standards and build tools.

We approach this by first dividing the task into visual, syntactic, and semantic components. These specifications can focus attention on the addressing requirements of the different groups. Additionally, these components rely on a structure for documents that incorporates existing library models and extends them to new modes of operation. Berkeley's MVD is one plausible structure which will allow prototype development and explorations.

1 Introduction

The issue of representing knowledge is deservedly murky, but certainly there is a general agreement that scientific documents must do this. One component to this encoding appears to be the representation of mathematics in documents in an unambiguous form. *Since one can easily show this is impossible in a general setting, we have to settle for something less.* Much less. For example, from the perspective of current commercial document scanning systems, one cannot expect to know whether something on a page is mathematics or not! Research projects to find "math zones" are only modestly successful.

Thus, even segmenting a page correctly with respect to text (whose knowledge contents we will ignore as being "not our problem") and mathematics (or other material) cannot be assured.

2 A diversion

Before throwing up our hands in dismay and compromising on our objectives, let's follow up this mathematical rigor idea a bit. If we can only communicate that which we formally represent, there are many world-wide web resources to look at suggesting paths to take before we can communicate. For example, on formal representations of many concepts you could check over the listings being assembled by R.B. Jones including a report on mathematics[1].

Or perhaps you should start with QED, whose stated, but as yet unachieved goal is to organize an "international project to build a computer system that effectively represents much of important mathematical knowledge and technique. The QED system will conform to the highest standards of mathematical rigor, including the use of strict formality in the internal representation of knowledge and the use of mechanical methods to check proofs of the correctness of all entries in the system. A principal application of the QED system will be the verification of computer programs. For background on the idea of the QED Project see the QED Manifesto[2].

On the other extreme, if we deliberately ignore this noble line of work and others like it, and assert the (in general false) hypothesis that the semantics for a piece mathematics is exactly its appearance, we can make some immediate short-term progress. Old mathematics is represented by scanned images, and new mathematics is fully communicated if we just agree on a single typesetting system. Since there are already a number of typesetting systems for mathematics (among academics, TEX and EQN are prominent), all that is needed is to pick one[3].

Of course one can argue that making such rigorous demands on the mathematical contents of a paper is unnecessary: We do not demand formal correctness or unambiguousness of text. Some text is incorrect, and ambiguity, especially in poetry and in letters of recommendation, may be deliberate!

3 Design Criteria for Scientific Document Systems

How can we meet the needs of the communities indicated?

3.1 Who needs what

Publishers usually demand full control of the printed appearance of a document. They reluctantly have surrendered some of this control but in the process may have saved money by coordinating with authors who provide "camera-ready copy." There are several common computer "batch" languages for typesetting that assist the user in specifying the appearance of mathematics, and may also impose a level of uniformity or style on the mathematics so as to be consistent with one or another journal's style. These languages (TEX, troff, XYvision for example) use positioning instructions, and therefore distinguish between 1/2 and $\frac{1}{2}$. They also specify, within the context of a global style specification and various local constraints, the complete presentation on the page.

Moving away from fixed paper as the primary medium, one must now consider that readers of a publication have different needs. While some will print it on paper, others will view it on a much lower resolution video screen, or perhaps in a different font. In these circumstances publishers and authors producing material for such display, perhaps on the world-wide web in hypertext, do not know about display size and print resolution and so place the reader (or his proxy, a network browser) in control of fonts, page-sizes, etc. That is, the *reading and displaying "browsing" program* rather than the *publishing or "sending" program* decides whether to display $\frac{1}{2}$ or 1/2, given some communicated representation of "one half". Or perhaps expressions are broken between lines in different ways.

Given this reader/writer choice, we see that we are already moving away from appearance-centric systems (where the primary point is to produce a particular appearance as a human-readable document on paper or display) to a more abstract form: *a structure for the information in the document.*

[1] http://www.rbjones.com/rbjpub/maths/index.htm

[2] ftp://ftp.mcs.anl.gov/pub/qed/manifesto

[3] And incidentally, to suppress all others. Evidence here suggests that perhaps the Spanish Inquisition had the right idea!

At least this kind of structure is needed for almost all other uses we have listed. The details of the structure are open to debate, but most thoughtful literature distinguishs at least two additional levels of structure.

One level which is generally called *syntactic* characterizes utterances by a grammar. In this way one can can distinguish "well formed formulas" from others. One could then legitimately require parentheses to be balanced, with matching sizes and styles. One can even make rules for breaking expressions between lines at operators of lower precedence. That is $a + b + c \cdot d \;=\; e$ should be split at the $=$, or at a $+$ rather than between c and d. This syntactic representation and its implications is widely believed in computer science circles to be well understood because compiler technology, including parsing of mathematical expressions, has apparently developed to a high art. This is unfortunately a false impression. The syntax of mathematics is far more elaborate than that supported by Fortran; even the far more extensible computer algebra systems like Mathematica or Maple flunk simple tests; Mathematica displays $\widehat{a+b}$ as though it were the perhaps quite different $a\hat{+}b$.

Representing math by context plus grammar can be formalized in various ways. Lavirotte/Poittier propose using context-sensitive grammars and disambiguation procedures. It is not clear how this would scale up in practice to large domains.

A deeper level of representation may require an agreement on what is generally called semantics. This constitutes a representation of "meaning" that can usually be mapped into one or more specific forms for existing programs, including typesetting systems. This semantic representation is not the same as any existing typesetting language because it has different concerns. While the typesetter must be concerned with inserting line breaks, more than likely an expression has no meaning other than "this is the encoding to produce an appearance". This skirts the issue that for many utterances, and especially mathematics, context alters meaning. For example, a semantic representation would distinguish between D_x an operator denoting the derivative with respect to x, and D_x, the x component of the D vector.

As an example of how one can separate syntax from semantics, observe that one might specify precedence so that $a \oplus b \otimes c$ means the same as $a \oplus (b \otimes c)$ but is different from $(a \oplus b) \otimes c$ by syntactic rules. Without specifying any semantics for $\oplus$ or $\otimes$ at all.

3.2 Documents, old and new

Humans who read and understand a scientific text are presumed capable of deducing the semantics of the utterance from the appearance[4] and disambiguating any notational problems. It would seem that an OCR program should attempt the same. This is a very difficult problem; some thought suggests that a full solution would be about as hard as building an "artificial intelligence" that understands all of mathematics. The skeptic must conclude that Legacy paper-only documents can cause problems with any document representation system: unless we are willing to re-key and encode old papers and thus also bear the possibility of introducing errors, we must record such documents as images. We agree that we must settle for partial solutions.

Setting aside for the moment the problem of understanding and encoding *existing archival printed documents*, what about those that are in some electronic form already? Most such existing encodings are aimed at producing the visual appearance only, and are not very helpful, semantically.

Can we proceed from this point in time forward and start encoding articles properly? Perhaps publishers or others could rely on the authors, referees, or editors of documents to inject the semantics using some agreed-upon language. This begs the question of the agreed-upon language, since that does not exist (yet). But if we had such an encoding we would also have a target for OCR/ document analysis programs.

New documents authored as electronic text do not have to undergo OCR or manual keying, so some sources of transcription errors are eliminated. Our expectations have evolved with regard to digitally authored documents. For instance, the ASCII fixed-width font literal representation

[4] Although it is not clear that a calculus student getting full marks on an exam really has to understand the semantics associated with each of the 7 symbols of the sequence $\int f(x)dx$ or this particular sequence. Frankly, explaining some of the meanings may be done best by a history lesson.

```
                                2
f(x):= a  + a x + a x
        0    1     2
```

is fairly easily read here but is targetted to a narrow range of paper-like display systems. But all kinds of manipulations confuse its sense: changes of font, margins, tab sets, pagination, conversion to Braille, cut-and-paste, etc.

We must aim for a much broader usage allowing at least embeddings and attachments. Embeddings provide cross-links from locations in a document model to other locations in the document (say, table of contents to pages, references to bibliographic information), links out of the document (file-system or network hyperlinks), and links to data attached in some way to the document. This data could be active mathematics, animations, sound recordings, image-maps, data bases, low-resolution approximations to images, approximate ASCII equivalents to text, suitable for searching, attributes (such as language, key words, etc.) Once we can include links to arbitrary computer programs and data, our options expand to offer the storage of "objects", "classes", "locations" etc. Various high-level notions of (say) interacting models can be hypothesized, and at least one technical base (namely Java programs) has a prospect of a broad audience. In the business community, several "standards" come to mind, OLE, CORBA,

A successful document-handling system should be designed to "interoperate" — to read and write unchanged all the parts of a document including those parts that cannot be interpreted by that system, but perhaps by some other systems. A plausible technique of last resort is to insist that any old document be convertable to the new standard by including it within some kind of "thin wrapper". Ultimately

for our criteria we are left with one, and it is hardly controversial:

Any future scientific document standard should be able to encode mathematics in a useful form. Examples of usefulness may include mathematical re-use of formulas such as evaluation of numerical expressions. A consequence of this is we must exclude efforts that support appearance-based mathematical typesetting, and cannot be extended.

4 Specifications

Growing from a handful of organizations interested in math representation, we can count a minimum of 18 now active in addressing problems of publishing math[5]. There are also a number of thoughtful individuals who have suggested solutions to some or perhaps all of our problems regarding mathematics. The specifications that have emerged to date tend to be of quite limited scope, just enough to get a particular job done. Some proposals are in direct conflict with others, some in inessential ways, but some in significant ideas. Some differences are exacerbated by financial and intellectual investments in old technology. By keeping an open mind and especially by being vague, each standard appears to hope to envelop the competition. Thus it is sometimes difficult to pin down the differences. Here is a current view of some front-runners.

4.1 SGML may help, but...

Although in its native form it is clumsy almost beyond comprehension, the possibility that SGML (Standard Generalized Markup Language) is explicitly poised to form a standard for the logical structural description of documents, even those including math, has evoked pride and (in the math-specific community) an element of dread—what if an unworkable standard is set up? Since there are continuing SGML working group discussions of many issues including mathematics encoding, with luck, these may converge to something resembling our suggestions here, (especially if I participate in them!), and one hopes they will be agreeable for other science/math-fans.

Aside: I suspect that SGML fans for other content-intensive material may also also have similar reactions; perhaps each group must develop a multi-layered elaborateness such as we are about to propose for mathematics[6].

---

[5] for a more current annotated list that I am trying to maintain, see `http://http.cs.berkeley.edu/~fateman/MVSD.html`

[6] An example of content-intensiveness: Why not have SGML optionally encode English

4.2 Divide Math into Three Parts

As we have intimated earlier in this paper, for mathematics, ideas that have been used recently seem to converge into a three-part representation. Depending on the context of usage, one, two, or all three may be needed.

Part 1. A visual prescription for typesetting a mathematical formula. An example of this is TeX. Most readers will be familiar with this or the essentially similar typesetting models in other script-based math typesetters (`troff`). For simplicity, the math mode in TeX uses a model that assumes the width provided on a page is adequate to lay out any formula it is asked to typeset. It does not compute break-points at which it can, at its own discretion, break a displayed formula over several lines. Explicit human intervention, knowing the available line-width and resolution, is necessary for breaking long formulas. The WYSIWYG formula typesetting is even more explicit in human intervention to lay out formulas, even when no breaks are needed!

Part 2. A syntactic or structural description. Although there are many details to be considered, our first goal should be to distinguish it from the strictly visual typesetting description.

One criterion that distinguishes structural from visual mathematics is that here for the first time one can (and must!) assign many different meanings to white-space and horizontal adjacency. For example, $(a+b)(c+d)$ might be understood as the product of two expressions (i.e. $ac+ad+bc+bd$), but a superficially similar form $(f+g)(x+y)$ might be encoded differently under the assumption that f and g are functions. In this case horizontal adjacency means function application, not multiplication, and the expression means $f(x+y)+g(x+y)$.

Another distinguishing characteristic is that two different forms may have the same simplified or "canonical" representation. For example, $\sin x$ is likely to have the same structure as $\sin(x)$, and $1/2$ and $\frac{1}{2}$ would be identical[7].

A typical unambiguous structural encoding is stored in a computer as a directed acyclic graph (DAG) but for human consumption this can be laid out a a character string. For this purpose we prefer the use of delimited prefix operators: the Lisp language provides this in a compact form. SGML can provide this too, but as typically specified, with a hugely verbose overhead[8].

Part 3. A semantic description. This is the kind of encoding that is needed for "math-aware" systems to operate on mathematical expressions: typical operations might be numerical evaluation of an expression, rearrangement of an equation to isolate a term, differentiation of a expression, specialization of an expression such as x^2+a^2 specialized for $a=2$ is x^2+4. Many issues arise in this "knowledge representation," and we choose only three that must be standardized:

1. Notation: As an example, what is the name of the logarithm function? log or ln. What is its base (10, 2, or e?). More substantively, how are the many kinds of graphical notations mapped into "functional" notation? Do we call it `log_base_e`?

2. Definitions and properties: Continuing our example, is that the real or complex logarithm? If complex, where are its branch cuts? (And in what language are they and the Riemann sheets described).

3. Specification of domains and contexts: Often, symbols and their structural arrangement are re-used. Variables may have restricted domains (z is complex, M is a matrix). Basic operator symbols such as $\cdot$ and $+$ for multiplication and addition are overloaded in common usage. Here they may need to

---

words and sentences with tags noting their meaning? These would presumably aid in translating such a document to another language.

[7]There might however be a "visual display hint" associated with structures if it were necessary to make distinguishing displays, as there obviously is in the representation of this paper! In a typical Lisp design, we would have `(sin x)` or `(/ 1 2)`. But one could imagine a hinted representation `( ''/-displayed-upright'' 1 2)` at some inconvenience in processing.

[8]An abbreviated SGML mode equivalent to, or more compressed than Lisp is certainly possible, though.

be disambiguated somehow if they are to be used in different context to denote operations of floating-point arithmetic, string algebras, vector or matrix arithmetic, etc. Some more elaborate domains might include, "polynomials with coefficients in the finite field consisting of the integers modulo 7 with indeterminate x, represented as a dense sequence of explicit coefficients"

4.3 Discussion

Typically a semantic encoding is an envelope around, and embroidery to a structural encoding.

Do parts 2 and 3 allow us to compute part 1? Yes, a generally-adequate visual encoding of simple expressions can be produced—as is demonstrated by those computer algebra systems which can take their internal form and produce 2-D typesetting or even TeX. Thus specifying the visual encoding for well-known kinds of mathematics is optional in that it can be generated on command. Even more, one could claim that some programs do a better job than some humans by being fast, accurate, and consistent in deducing reasonable line-breaks for large expressions or making other efforts to display data in a useful format. But outside the well-trod areas of their expertise, and even inside these areas, automatic TeX generation is a delicate matter. Simplicity in display is not simple. Some computer algebra systems prefer $1 + x$ to $x + 1$; we prefer the latter, most of the time.

Sometimes an author wants to display some kind of tableau, or has a particular point to emphasize by means of size or alignment or ordering of terms. This does not fall out of routine structural encodings. While a continued fraction might be automatically typeset as

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \cdots}}$$

a careful author might much prefer (and can specify in TeX) this display:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cdots}}.$$

To illustrate a potentially *very common failure* of this structure implies display is where a computer algebra system simply does not contain sufficient notation, and it has to be custom built. Consider an expression I used in a recent paper to denote certain linear combinations of trigonometric terms:

$$\sum_i p_i \left\{ \begin{matrix} \sin \\ \cos \end{matrix} \right\} (a_i x + b_i y + c_i z).$$

This "alternative function" notation for sin or cos (choose one), is understandable once explained, but is one of an infinite variety of potential notations one could make up that is missing from some predefined system. A program with an extensible structural to visual mapping helps, but may merely be a temporary substitute for a semantic solution.

Does part 2 (structure) allow us to compute part 3? In fact, most computer algebra systems use some version of structural encoding, e.g. in Lisp this might mean $a + b \sin x$ encoded as a data structure that prints as `(+ A (* B (SIN X)))`[9]. These systems then carry on many manipulations rooted in semantics simply because they have a built-in default model that maps structure to specific semantics. If that model is correct for the operation in question, the program may work. For example, one program dealing with that structure probably assume a and b are stand-ins for real constants not dependent on x, a real-valued variable, and that the symbol $+$ means real addition. But another program might assume a domain of complex-valued variables for a, b, and c. Thus part 2 implies a subset of part 3. This is generally far too restrictive, and can hardly be used for real-life mathematical publishing. Two apparently identical structures have different meaning in different contexts!

[9]this might also be rendered in some SGML schema.

What if you only have part I (visual appearance)? This is what humans see when reading a scientific paper. Algorithmically, however, it seems that the reader combines a fair dose of knowledge and/or a very simple expression along with a set of default model assumptions to parse non-trivial expressions. The proof that it is possible is that humans do this all the time. The proof that it is hard is that they often make errors! Success with $a + b \sin x$ does not necessarily generalize, as anyone who has puzzled over a mathematical expression knows.

If you look at serious articles from journals in applied science or mathematics with the view to encoding them—without knowledge of the material—you run into a dead end: visual appearance when taken out of context is not a reliable key to parts 2 and 3.

To avoid all problems you must either develop a super-human mathematical intelligence, or invent (or discover) structures and new semantic models for part 3 and 2, relating these to the visual notation.

A natural question is: *Isn't semantics the whole ball of wax?* What is the role of the syntactic component? Basically, even if you have tacit or explicit understanding of the names and operations in context, computing with them still requires some data structure to use as a skeleton for this information. A natural method for doing so is to place tags on a structural representation. There is a long history of annotating trees in computer science, and this history may even provide guidance in programming. A more sophisticated computer science approach would place such tags not in the data objects (linked lists or trees or nodes) themselves but as associated structures. That is, a visual hint might be among the attributes of a class of objects rather than individual objects, or perhaps it would be an attribute of a class at some level in a class hierarchy. Conceptually one would ask an algebraic tree to "display yourself on a 300dpi printer" and appropriate hints would come into play.

5 Practical matters

How should these three parts be encoded?

This is a matter of standard-setting and pronouncements, and is partly a political and economic question. Who should guide this? At this time, there is sufficient disagreement that legal standards will not emerge in the short run, but some de facto standards seem to exist as required for effective communication.

There are several widely used visual text/math typography standards. This paper uses TEX, a popular academic system. Frame, XYvision, and a few other systems are also in wide use for math, and many other programs such as Microsoft's Word have assoicated mathematics helper applications.

The opportunity for a de facto structural and semantic standard is more open: if anyone were to publish a substantial body of scientific material in a nearly satisfactory form, then the many users of this material would find it advantageous to promulgate this. For example the Abramowitz/Stegun [] handbook, although now somewhat dated, standardized the definitions and names of many special functions, and listed their properties. A similar reputable new standard, perhaps a revision of this volume with appropriate electronic links, could analogously form a basis for some semantic and structural representation. While this presumably would not manage to cover all of mathematics, such a revision would provide a new plateau of applied mathematics and special functions. It would not have legal status, but an endorsement from the US National Institute of Standards and Technology might be helpful.

In the absence of such a fount of wisdom or any other acknowledged authoritative source for structural standards, we fall into the amiably warring camps of various design proponents. All of them claim compatibility with each other, but each continues to pursue a separate course, hoping, by vagueness, to engulf the others. Two such groups are developing Openmath and Math SGML. The internal forms of computer algebra systems such as Maple, Mathematica, Macsyma, etc. are credible structural models, although (inevitably) incomplete semantically, and (at the moment) incomplete with respect to typesetting hints.

For semantic mathematics beyond that which is implied by the structural approach plus "default semantics" there is very little present technology in use. Probably the most advanced currently in combining domain semantics and manipulation, is the computer algebra system, Axiom, especially in extensions of "classical" modern algebra.

The advantage of starting with a computer algebra system, if it is possible to do so, is that there is an enforced agreement on structural and associated semantic representations. If the coverage of such systems were adequate[10], the "notebook" documents that are composed by users of computer algebra systems would be credible solutions to the future encodings of mathematics. It would take a substantial leap of faith for any serious mathematics publisher to rely on any of the current systems notebooks as represented by, for example, Axiom, Macsyma, Maple, and Mathematica. In spite of their competitive stances, these systems are relatively similar with respect to their representation, making exchange of data plausible, if not immediately obvious.

Two products that are more document-centric yet have strong ties to computer algebra are perhaps more plausible, in that advanced math that cannot be manipulated can still be represented structurally and visually, are IBM's Techexplorer and TCI's Scientific Workplace.

6 Putting the pieces back together

Advocates of each approach most often view their own solution as sufficiently general as to encompass anything of interest, and *a forteriori* strong enough to encompass anything anyone else is proposing. A proposal need not be explicitly more encompassing if it has sufficient extensibility.

We should proffer our own, developed in the UC Berkeley Digital Library Project, known as Multivalent Documents (MVD). In brief, the relevant aspects are as follows.

- A document ordinarily has a presentation view that can be printed or read from a screen. This is a graphical, bit-map, or similar representation. This view might be video or other "original" portrayal of information.

- For those items that are partially or completely text, another view is available consisting of a tagged OCR layer. This view can be revealed by asking to see the whole page in ASCII, or it can be observed in pieces and indirectly y placing a "lens" somewhere on the presentation view. This shows an ASCII approximation of the text. Because there is a keying between the OCR and the presentation, operations that make sense (searching, sorting, substitutions) on the ASCII OCR, can actually be simulated on the bit-map picture as well. One can search for words in the "picture" by really searching in the OCR layer and they mapping back to the presentation. Target results can be highlighted graphically. For another example, sorting a "picture" of a numerical table is possible.

- For forward or backward references within a paper, or to out-of-file reference, html-like links can be supported. Underneath a reference we can place (for example) a BibTeX entry.

- For pages that have mathematics, additional (optional) layers can be constructed with contents that consist of the visual, syntactic, or semantic representations. They might be computed automatically, or, given the current state of the art, more likely created by human insertions. Our experiments of the last year or two include prototypes for these layers:

  1. a Lisp-language positional (syntactic) description of the mathematics in that zone.
  2. a TEX language utterance that could be reprocessed to approximately re-create the appearance of the mathematics. This is derived from the previous description.
  3. a syntactically legal Lisp language equivalent suitable for direct manipulation in our local computer algebra environment, or in principle exportable to Mathematica or Macsyma programs. We have not defined a semantic layer other than implicitly by this simple syntactic layer plus the context of the computer algebra system.

[10] Coverage is likely to be adequate only for relatively simple mathematics

A convenient user interface for this multi-level view is critical if it is to be successful in providing wide access to the contents of such an object. Here is where a critical technical decision has contributed to the impact of MVD in applications.

The interface is defined in Java, making it available, downloaded at run time, to any World Wide Web browser. Thus the access to the MVD views is available to nearly every person accessing the library. Users not need any special hardware equipment or sophisticated software setup beyond that needed for "web access". The MVD usefulness does not depend on plug-ins that have to be pre-loaded, nor does it depend on a choice of browser, operating system, or hardware (other than all support Java capability).

How might this work in practice in the future for scientific documents?

The top "presentation" layer can be in fact what we are using now (GIF for screens), or perhaps a more scalable presentation layer like PDF. The lower, more hidden, layers of scientific documents can include reference layers, cross-link (html style) layers, links between tables-of-contents to sections, and encodings of mathematics, chemistry, biology etc.. Note that we have remained flexible on the encoding; anything that is proposed for encoding math that can be linked by its geometrical coordinates to a visual section of the presentation page.

References

The most useful references are on-line to documents undergoing frequent revision. Active international email discussions have recorded the ebb and flow of ideas. Web search engines may find even more activities, but I have tried to link ones I've found informative together to this page:

`http://http.cs.berkeley.edu/~fateman/MVSD.html`.

Appendix on Typesetting

As an example of how one might encode the continued fraction example of section 3, consider the linear form `a[0]+1/(a[1]+1/(a[2]+...))` as it might be typed in some linear computer language. This which might be represented in a structural form, for example, in a Lisp-like notation:

```
(+ a0 (/ 1 (+ a1 (/ (+ 1 a2 cdots)))))
```

or in a TEX form as

```
a_0+{1\over a_1+ {1 \over a_2 + \cdots}}
```

or a better TEX form

```
a_0+{1\hfill\over\displaystyle a_1+ {\strut 1\hfill\over\displaystyle a_2
+ \cdots}}
```

or as a semantic form (here we have made up the format here, keying everything to an operator "CF" for continued fractions):

```
(CF a0 a1 a2).
```

Naturally the semantics for CF must be agreed upon. In this case, all the details could be generated from the semantic representation, although a particular instance of use might require some indication of variations in the preferred display.

We can anticipate that in general we would need a multiple-part representation: A bare-bones structure for $x^2 + 3x + 1$ might be

```
(+ (expt x 2) (* 3 x) 1)
```

but if we know more about it, it might look in a combined form like

```
([+ power-series, integer coefficients,
univariate, square matrix, [X boldface]]
```

```
(expt X 2) (* 3 X) 1)
```
and a more "informative" display might be

$$1 + \mathbf{X} + 3\mathbf{X}^2 + \cdots$$

Here we have recorded details of the semantics, specified (by saying it is a power series, and that X should be in boldface) how to print it, and the internal structure, displaying the particular coefficients 1, 1, 3.

In the interests of brevity, we have skipped over many of the details of intertwining these three components; this is properly the work of groups attempting to provide workable standards. Among many individual and group participants, recent cooperation among the OpenMath group, the SGML Semantic Math group, and the W3 consortium may bear fruit.

Appendix on Ambiguity

We have found that searching through source material provides a nearly unlimited collection of ambiguous mathematics, and that it is surprising how few people who propose universal descriptions of mathematics appear to have looked at the literature. If one were to come up with a universal grammar for mathematics that was sufficient to cover integral tables, that in itself would be a challenge.

The first kind of ambiguity is when the same notation changes meaning over time or from publisher to publisher, or even from section to section in the same paper! In one historically significant table of integrals, Bierens de Haan's [?]. we found the use of "notation de Krampf" which looks like $c^{a/b}$ and means $c \cdot (c+b) \cdot (c+2b) \cdots (c+(a-1)\cdot b)$. In particular, $1^{a/1} = a!$. The table exhibits expressions like $1^{a+n/1}$ which appear to mean $1^{(a+n)/1}$ or more conventionally, $(a+n)!$. Clearly, merging results in this table with later publications would require recoding.

Another area of concern is knowledge of particular *names* or functions: $J_1(a+b)$ could be the product of a symbol J_1 with $a+b$ or more likely in the context of mathematical analysis, or physics, it is the application of the J_1 Bessel function applied to the value $a+b$. A computer algebra system would disambiguate the cases by other syntactic or name clues: in Mathematica, `J[1](a+b)` vs. `BesselJ[1,a+b]`; in Macsyma, `J[1]*(a+b)` vs. `J[1](a+b)`.

One would expect that at least there is common agreement on the position of exponents, but: $\log x^2$ could be $\log(x^2)$ or $(\log(x))^2$. The form $\log^2 x$ could be $(\log(x))^2$ or $\log(\log(x))$. Given the usual convention on the use of Roman font for operations like log, it would be unlikely to mean $(\log^2) \cdot x$. (Since $\log(x^2)$ may be simplified to $2\log(|x|)$ or perhaps just $2\log x$. One would normally not see $\log x^2$ in a mathematical paper with this meaning.) But notice that reasoning based solely on position will not be sufficient. The expression fx^2 is either `f*x^2` or $f(x^2)$ or $(f(x)^2)$. But by the reasoning given above, if $f = \log$ we would not ordinarily allow the first of these interpretations: `log * x^2` is unlikely.

Although they would appear to be a related geometric relationships, the notations $\tan^{-1} x$ and $f'x$ are nevertheless usually quite distinct. Denoting inverse functions by a negative exponent may be more "cute" than systematic[11]. Denoting derivatives or other related functions by primes can certainly lead to confusion. $f^{(2)}(x)$ for a second derivative is troublesome.

Just to show that computer typesetting has not solved all these problems.

Appendix MSVD.html

An extract from the current version of `MVSD.html` as of July 30, 1997.

```
This is a supplement to a
talk  presented at 4th ICDAR 97 (Ulm, Germany).

If you look up only one place on the internet, an approach that we
think is fundamentally right is <A HREF=
"http://dave.cs.cornell:8002/Simlab/document.html"> the SIMLAB project
at Cornell</A> using a semantic and structural approach for
```

[11] Is $\log^{-1} x = \exp x$?

mathematics, designed for collaborative engineering and design work. The treatment of the context and overall viewpoint of communication known as Mathbus is described in this paper on <A HREF= "http://www.cs.cornell.edu/Info/Projects/NuPrl/documents/colmath/it.html"> collaborative mathematics environments</A>. However, no one has a ''lock'' on the exactly right approach and this is not so different from some of the other contemporary projects.

Next, a diversion. If you buy into the rigor argument (namely that we can only communicate that which we formally represent -- and this is not a bad argument when it comes to mathematics), there are many WWW resources to look at before we can communicate. For example, on formal representations of many concepts you could check over the listings being assembled by R.B. Jones including this report on <A HREF= "http://www.rbjones.com/rbjpub/maths/index.htm"> mathematics</A>.

Or perhaps you should start with QED, an "international project to build a computer system that effectively represents much of important mathematical knowledge and technique. The QED system will conform to the highest standards of mathematical rigor, including the use of strict formality in the internal representation of knowledge and the use of mechanical methods to check proofs of the correctness of all entries in the system. A principal application of the QED system will be the verification of computer programs. For background on the idea of the QED Project see The QED Manifesto."

If we deliberately ignore this noble line of work and instead work from the (in general false) hypothesis that the semantics for a piece mathematics is exactly its appearance, we can make some immediate short-term progress. Here's an early paper that sets down what had by that time become fairly obvious. Namely that with this hypothesis, to communicate mathematics electronically in some standard way that could be understood by all, we just need to agree on a single typesetting system. There were already a number of typesetting systems for mathematics (among academics, TeX and EQN were prominent), Pick one. This is conveyed in a 1987 memo from the Network Working Group (presumably ARPA funded in some way)

<A HREF= "http://andrew2.andrew.cmu.edu/rf/rfc1003.html"> Issues in Defining an Equations Representation Standard by Alan Katz of USC/ISI. Katz comes up with a recommendation for an EQN ASCII "directive" system which allows simple inclusion of math-display objects in email etc. Actually, I am mischaracterizing Katz: in fact he argues that semantics would not be represented because the person typing the math may not even know what it means. This situation may have changed somewhat since 1987, with more scientists typesetting their own papers!

Another early paper is "Towards an Equation Exchange Standard" Task Group Agenda by Tom Libert, Michigan EXPRES Project (libert@citi.umich.edu (1987)). Libert proposes the task force address the question of making a ''content architecture'' versus just equation formatting, and should the group ''develop such an architecture, or simply wait for ANSI/ISO to do it in 2-3 years?'' Clearly the wait has not achieved the architecture, although arguably several other groups have tried to pick up the ball.

A subsequent public meeting at a SIGGRAPH conference concerning this topic of representation of equations resulted in a report by Dennis Arnon of Xerox, <A HREF= "http://www.es.net/pub/rfcs/rfc1019.txt"> Report of the Workshop on Environments for Computational

Mathematics</A> The key idea being proposed here is that an ‘‘abstract syntax’’ be defined for any mathematical expression, roughly corresponding to a functional notation (or equivalent). Using this, a SMS or Standard Mathematical System, consisting of a collection of SMCs or Standard Mathematical Components communicating via the abstract syntax run over a ‘‘wire’’ between them Typical SMCs could be assembled for use in one or more of four ways: Editing, Displaying, Computation, and Documentation.

This proposal is a useful clarification, but does not go far enough in elucidating how separate meanings could be attributed to the same abstract syntax in different contexts.

A more or less linear extrapolation from this position, but with the injection of SGML notation, is the current (May 1997) draft of the WWW consortium <A HREF= "http://www.w3.org/pub/WWW/TR/WD-math/"> MathML Mathematical Markup Language, an <A HREF= "http://www.w3.org/pub/WWW/TR/WD-xml-lang.html"> XML (Extensible Markup Language) application for describing mathematical expression structure and content. I think this is a reasonable document still undergoing revision by a small group of apparently sensible individuals. I believe this WWW consortium activity originated from the easily recognized need for adding visual math markup to HTML; it is not clear if this is going to be the same as the results of the <A href= "http://lists.w3.org/Archives/Public/w3c-math-erb/msg00347.html"> SGML Math Workshop Group </A>. and the HTML-Math activity at WWW Consortium. (HTML-Math seems to differ from SGML math in that it is explicitly stated as a goal that it is easy to learn and edit by hand!) More chatting of alternatives to this kind of encoding are discussed in the minutes of a 1996 SGML Math workshop held at the University of Illinois, as part of its digital library project on ‘‘Federating Repositories of Scientific Literature.’’

Perhaps all of this will converge with the OpenMath group’s work, which I see as following the notion that mathematics must be made formal and explicit based not on axiomatic set theory and logic, but on some rigorous combination of "classic" modern algebra, some applied math, and a strong dose of programming language design. The OpenMath working group is now working on ‘‘Content Dictionaries’’ expressed in a version of SGML. You can view them by following links from the home page. The success of this effort is predicated on the cooperation (and funding) of participants interested in a grand programme of automated mathematics. It is reminiscent of the the QED Manifesto in this regard. Nevertheless, there is an overlapping cast of characters for all the groups mentioned in this paragraph; most are listed at the previously noted site HTML-Math.

There are computer algebra systems with notebook-like front each, each of which would presumably be delighted to see its own technology adopted for the representation and computation of mathematics. Even the most ambitious of these today would have to be extended in semantic scope before being using for computing in all the areas that can be (easily) typeset. Here are a few systems with Notebooks: Mathematica, Maple and Mathview, Macsyma, Axiom which is supposed to link nicely with IBM's TechExplorer and MuPAD, a research project at Paderborn University. There are also broad interests in computer algebra and display at Project SAFIR at INRIA (France). These and other computer algebra systems or projects are mentioned with links in the SymbolicNet listing of systems.

An ambitious project that began as a communication protocol design for distributed scientific computing, the MP or Multi Protocol at Kent State is designed to support efficient communication of mathematical data between scientifically-oriented software packages. MP exchanges data in the form of linearized annotated syntax trees. At a level above the data exchange protocol, dictionaries provide definitions for operators, symbolic constants, and annotations. Binary encodings are used.

There are a variety of one-off projects like <A HREF= "http://www.lfw.org/math/top.html"> the MINSE project</A>, with a "polymediator" which is a computer program available to convert, on request, pieces of readable text into urls of stuff. Or <A HREF= "http://www.math.ethz.ch/~shared/emb/GeneralFiles/Euromath.html"> the Euromath project</A> funded by the European Mathematical Trust, to ``enhance research support for European mathematicians'' by assisting in communication, apparently resulting in an editing system for mathematical publishing, and a bulletin.

If your primary interest is simply to display math on Web pages, then the Minnesota Geometry center would like you to use their WebEQ software which requires the user of a browser to download programs (UNIX or Windows) but would then process HTML-Math and other, TeX-like stuff. Poliplus software's EqnViewer comes with a java applet that can be used to display pieces of math. Since their demo page hasn't worked reliably for me, I haven't been tempted.

 TCI Software Research has a selection of word processing programs including Scientific WorkPlace, which is a word processing system designed especially for preparing technical documents, with substantial support for mathematics. Options for this product include links built in to the Maple or Mathematica computer algebra systems. SWP represents an alternative route to the ``notebook'' paradigm that seems to be a result of convergent evolution in the Maple/Mathematica (etc) battle. In this case the evolution has come from an almost-WYSIWYG almost-TeX-intermediate-form word processor.

There are other products described in links from its home page.

A nicely working program, but ONLY for windows, is IBM's TechExplorer which deals with TeX code and must be down-loaded to your computer as a plug-in. Simon Fraser University has interesting Web-based mathematics presentation and computation facilities in their Organic Mathematics Project. This project demonstrates alternatives to the notion of a static, even hyperlinked, textbook document. People interested in a more multi-media approach to math should find this interesting.

How about publishers? Consider Lightbinders, a producer of CD-ROMS for scientific documents, including the new 5th edition of I.S. Gradshteyn and I.M. Ryzhik's Table of Integrals, Series, and Products (Academic Press). Naturally. a form of encoding was needed for this, but sad to say, only appearance was encoded. This uses Dynatext from EBT (now INSO), which allows either SGML or TeX format and is compatible with ISO TR 9573 part 11 DTD, the AAP DTD, and the ArborText DTD. In fact, all forms are converted to TeX, then DVI and then rendered as raster images. The advantage of starting with SGML is that search tools that have been adapted for other SGML forms might be able to search in SGML-math too. Searching in TeX is not supported <p> Among the companies bridging the content-production tools and publishing industries we also include Xyvision, an electronics based publishing company used by some scientific journals, and ArborText, which provides SGML-based editing and publishing software. Another company in the marketplace supplying electronic publishing for the scientific and technical markets is <a href="http://www.az-tec.com" Aztec Corporation. Their concept is to provide platform-independent CD-ROM based programs using their Intelligent Manuscript Architecture combining formatting, search, and problem-solving. A sample (on the Web) handbook of differential equations is on exhibit.

After looking through this material, you might wish to peruse Roadkill on the Electronic Highway: The Threat to the Mathematical Literature for an argument that a distinction must be made between the more casual electronic publication and ‘‘real’’ publication.

<h3>The real world </h3>

The issue of communicating ‘‘objects’’ of any stripe is not unique to mathematics, and is simply a special case of what many people believe computers should routinely support. Indeed the flexibility of computer networking should allow us to connect almost any digital information source to any information sink.

As a non-mathematical example, a human user interacts with a graphical user interface (on a client computer) to get access to a data-base server such as a library catalog system. There is a necessary intermediary function here: the packaging up of a query for delivery, as well as the packaging of the response. Not all sources and sinks have the same data interface, and there is a lot that can go wrong with the middle. For example, some particular server may be busy or out of operation, but maybe there is an alternative. Intermediation is the realm of a new software category of Middleware tools. The vendors of middleware tools claim to assist one in packaging up any old thing and deliver it anywhere else. For greatest utility, it has to be done in a machine independent fashion. Component technology for linking machines can be based on any of a number of standards, and CORBA (Common Object Request Broker Architecture) is one worth mentioning.

In fact, if we compare the objectives of CORBA (which also has task forces, such as financial, multi-media, manufacturing, etc) we find material like this:

<p> The Manufacturing Domain Task Force Mission: <ul> To foster the emergence of cost effective, timely, commercially available and interoperable manufacturing domain software components through CORBA technology <p> Recommend technology for adoption that enables the interoperability and modularity of CORBA-based manufacturing domain software components <p> Encourage the development and use of CORBA-based manufacturing domain software components, thereby growing the object technology market <p> Leverage existing OMG specifications <p> Recommend liaison with other appropriate organizations in support of the preceding goals.

If we substitute (for example) OpenMath for CORBA and OMG, and mathematics for manufacturing, we have a similar story.

A currently non-CORBA approach to the same problem which is based on Java, is RMI or Remote Method Invocation, which provides a plethora of facilities intended to make Java-encoded data (presumably including programs) usable and transportable between platforms. Conveying structured data described as elements of a class is intended to be entirely transparent between the sender and the recipient. If that data consists of ''math'' objects, then maybe we have some software technology waiting for our use. This assumes that we all agree on the form of such an object (and that kind of standardization is most of what we are concerned with above). RMI also assume we are either using Java or adopting a solution that is Java compatible. Furthermore, it makes the reasonable assumption that we are dealing with objects that can be serialized. This means that they can be written out as a serial byte stream which can be read back into another virtual machine running Java as instructions to create the "same" object. (Common Lisp does this free with its print/read programs, for example).

If we do not agree on an object description, translating among variants would be needed. If we want to search for translation software, we might find the software vendors who providing component exchange technology such as Lotus's InfoBus.

This kind of technology -- embedding the translation process between the front and back ends of programs -- seems particularly unappealing compared to abiding by a standard. Nevertheless, in the absence of agreement, this approach of translatation is another position of cooperation among diverse programs.

<h3>Example of an Ad Hoc Solution to search </h3> {As an aside, searching for mathematics may end up using the ad-hoc techniques used for searching in technical chemical databases, in which case one can look at <A HREF= "http://www.cnidr.org/ir/stas.html"> the scientific and technical attribute and element set (STAS)</A> maintained by CNDIR, the Clearinghouse for Networked Information Discovery and Retrieval. This working document does not (yet) deal with mathematics, but shows a one direction for introducing a search standard. It is in fact oriented toward denotating chemicals as is needed for standardized library queries using the Z39.50 network standard protocol. Field indexes are available for denoting abstract (62) to zipcode 2794). Also included are the prespecified index numbers for things like OSHA (US Occupational Safety and Health

Administration) permissable exposure levels (3910) or vapor pressure (2973), or the somewhat mysterious Uniterm (2790) "Code and text which represent general concepts, chemical fragments and chemical substances in the IFI files". Not everything in Z39.50 is chemically related. Though the breadth of its coverage appears huge, it seems it is still growing. If you look at some details in the Library of Congress promotion of this standard you will see that there are registered record syntaxes for forms like postscript, html, gif and sgml. For example, the initial assignment to sgml is 1.2.840.10003.5.109.9. It helps put your standards into perspective.

7 Optional Digression on why Mathematics is a black hole, and how Computers may or may not help

Historically, the idea of a fixed formal language dates from the Vienna (until 1933) Circle of Positivists. Interest in formal languages increased with the advent of computer languages. We were suddenly made aware of the real requirements. Computers reject the mathematician's hand-waving, ambiguity, and eternal language -changing. No longer is it adequate to formalize just the foundations.

"At present, we teach by intimidation and learn by imitation. The mysteries are repeated in a raised voice. At last, the student gives in. But computers are immune to such treatment."

[deBruijn, automath, 1973: Automath, a Language for Mathematics]

deBruijn hoped to provide a formal language that could be used for stating proofs. This language could be processed to confirm the steps in the proof. He explicitly suggests that automath might give risedto interesting new theorems, computerization of "a general mathematical library (dial a theorem?) improved publication standards and higher levels of man-machine interaction."

It is nearly 25 years later. Computers are far more powerful. Where did these ideas go?

Some were incorporated in later systems, Mizar for example, and perhaps in Gordon/Milner/Wadsworth (1979) Edinburgh LCF: A Mechanised Logic of Computation, Springer LCS 78. Nederpelt et al vol 133 of Studies in Logic and the Foundations of Mathematics, North Holland (1994).

refs: see http://www.rbjones.com/rbjpub/logic/jrh0107.htm www.abo.fi/ jharriso/

From a philosophical and logical point of view, one might insist that only that which is completely formally specified can be used as the basis of understanding or computation. Certainly computers are notorious for not understanding what they are not told! Yet it is commonly believed, in spite of various setbacks in constructivism, that with sufficient effort, all of modern mathematics can be placed on a solid footing Furthermore, the appearance of new designs for computer languages and fast computers we have breathed new life into efforts at completing such formalisms. (cite QED project, HOL)

From a practical point of view it is unreasonable to require formalization of all prerequisite mathematics to some utterance prior to communicating it to another person (or program). Only the dullest pedant would teach a subject without first providing some perspective.

All mathematical publications rely on a vast common human mathematical context, and most require some more specialized context.

To the extent that a reader does not share the context – the notation, the basic assumptions, or even the intellectual interest assumed by the writer, miscommunications can occur: missing context from technical papers can render them gibberish to all but the most adept specialists.

Of course it would be wasteful if specialists formally restated the foundations of their mathematics in every paper. Some authors provide preliminary or background material, some provide references to establish notation, and some assume that the reader has all necessary background without explanation or apology.

Arithmetical and mathematical communications among computer programs can rely on a set of basic *abstractions* which CAN be axiomatized and formalized. For example, we need not know the bit arrangements, that is, the *representation* in a computer word to deal with the concept of "integer between -2^{31} and $2^{31}-1$".

We can even deal with the abstraction "integer of arbitrary length", using multiple words of a computer. We can do so even though are integers too large to be explicitly written out in the memory of any computer we can build.

From the primordial mud of mathematical foundations, researchers have made careers out of building ever-higher levels of axiom-based computationally plausible algebraic systems.

There is a large gap between what has been achieved by systematic treatment of mathematics, and what is operationally or notationally possible. That is, nearly all computers have operational floating-point computation units that conform to the same IEEE standard. These units do not check their results against theorems; they produce results in conformance with theorems as a consequence of the computer design and implementation. A complete axiomatization of floating-point arithmetic, still something of a challenge, would not have much effect on most practice.

More plausibly, one can usefully consider that two copies of the same computer software library do not have to deal with axioms if they are cooperative in a task to (say) "find eigenvalues." The software defines the task by the programs, and since they are operationally the same, there is only limited opportunity for misunderstanding. Yes, it begs many questions as to what is meant by eigenvalue — how accurate are the eigenvalues? how are they ordered? Do we need them all to the same accuracy? Do we need only the largest or smallest? Our restricted context in effect answers all these questions by computing whatever is computed!

Certainly in applications (publications in applied science or computer programs solving a particular class of problems), humans generally operate with appeals to intuition or physical principles. Even in what would appear to be formal symbolic mathematical discourse, appeals via examples, numerical approximations, or pictorial geometric arguments are sometimes used.

...... This material above may seem a major digression (I think it is!) but the point is, if we are to talk about storing and retrieving documents we can work with them at several layers of abstraction running from

"Here is a data file consisting of a collection of ordered pixels"

to

"Here is a representation of (all) the knowledge in the document presented in a form that is easily processed (for example, by computer) in any of a collection of well-understood ways".

If we are dealing with scientific documents, the versatility of the representation directly affects the nature of processing possible. Except for the most basic and pedantic documents, a general computer program attempting to encode a mathematical document can be plagued ambiguities of notation and semantics, missing contextual information, and other problems.