Enabling Large Scale Coherency Among Mathematical Texts*

Stuart F. Allen and Robert L. Constable
Dept. of Computer Science
Cornell University

Abstract

Mathematical and program-code text is unique because significant portions of it can be anchored to counterparts in formal logical theories that are implemented by computer systems. These systems check formal proofs for correctness and trace logical dependencies among assertions. When elements of expository text, such as definitions and theorems, are formally linked to their implemented counterparts, we call the texts *semantically anchored*. Such texts exhibit considerable depth and authority.

It is possible to leverage substantial investments made by governments, research laboratories, corporations, and universities in creating large collections of computer-checked and interactively-generated formal mathematics, making this research investment, these collections, accessible to an extended community of authors, researchers, students and teachers involved with mathematics.

We advocate extending common authoring tools (text editors as opposed to formal proof development tools) so that they can easily produce semantically anchored documents suitable for dissemination along with the formal mathematics to which they are anchored; some texts would be newly authored, while others would be static text-based resources improved by anchoring. These tools will enable authors to create these documents by drawing on a large already existing and growing collection of formal material.

We expect that anchored documents will enable interconnected collections where the computers support exact common reference among concepts and thus greatly facilitate collaborative contributions to online collections and provide large-scale coherency among mathematical texts. We discuss efficiencies expected from such anchoring in formal material.

*This work was supported in part by the DoD Multidisciplinary University Research Initiative (MURI) program administered by the Office of Naval Research under Grant N00014-01-1-0765, and by the National Science Foundation under Grant 9812630, and by the National Science Foundation under Grant 333526.

1 Introduction

There are internationally established groups of specialists [26, 9, 34, 16, 21, 8, 44, 43, 23, 39, 35, 48, 12] who develop extremely precise formalized mathematics to a very high standard of correctness, using a variety of methods. There is already a significant body of such mathematics and, increasingly, it is being made available through the Web.

We advocate enabling the vastly larger numbers of readers and authors who are not among those specialists to take advantage of this body of mathematics. We are motivated by the desire to see a mutual support develop between formal mathematics collections and informal expositions. One obstacle to this goal is that prospective authors of expository texts or annotations will not generally be people who work within the communities of specialists that produce the formal material itself, and cannot be expected to adopt the specialists' tools; indeed, they may be unaware of the existence and nature of formalized mathematics collections.

The key is creating tools that allow nonspecialists to author texts that refer to specific concepts and entities in repositories of such formalized material. These formalized references serve to *semantically anchor* and implicitly coordinate diverse informal texts according to their common references to formal entities.

It is a practical necessity that these prospective authors be able to continue using essentially the same authoring tools and media to which they are accustomed.

1.1 Mathematical Text

Mathematical text is distinctive because in principle all of the definitions, claims, theorems and proofs can be expressed in a formal language that can be processed by computer. Indeed, what we mean by *formal mathematical language* is exactly that language which can be processed by computer in such a way that syntax can be checked, type assignments inferred, proofs checked and built interactively and parts generated fully automatically. In addition, algorithms that define computable functions can be evaluated, algorithms and data types can be mapped into those of existing programming languages, and symbolic expressions can be transformed. Collections of formal mathematics can be displayed using Latex, MathML, HTML, and other display software.

Over the past two decades computer systems have been built which can accomplish in practice what we have known for nearly a century could be done in principle. The systems dealing with proofs are called proof development systems or *interactive theorem provers* ("provers"), and those which process symbolic expressions are called computer algebra systems ("CAS"). Among the provers are Alf, ACL2, Coq, HOL, Isabelle, MetaPRL, Mizar, Nuprl, Omega, and PVS. Among the algebra systems are Mathematica, Maple, Axiom, Reduce, and SAL.

These computer systems have been used to solve scientific problems and to improve system design and software reliability. In this process, large amounts of formal knowledge have been created. We think that this large and growing collection of formal material can be of great value to parties besides those for whom the materials were originally produced

— even if the computer systems that created it are not easily accessible. The value arises in four ways.

First, the body of formal knowledge is correct and authoritative to extremely high standards. Achieving those standards by the ordinary social process of having many professionals use the material is very costly and inhibits rapid innovation compared to the cost of producing the formal text. (This is illustrated by the NIST effort [33].)

Second, the body is coherent and consistent in every detail. All the theorems and definitions *interoperate.* This assurance is something that authors find extremely difficult to achieve. Computer-enforced conformance achieves this at the cost of having expert users of these systems, people we call *knowledge programmers.* Appropriate technology could leverage the efforts of these people a thousand-fold or more.

Third, by including the formal material in documents, there is a way to guarantee points of *exact common reference* among all documents that share the same formal material. This is computer-checked common reference. (A large-scale effort at encouraging common reference across mathematical texts is the OpenMath Society[1], which has developed standards [4] for representing mathematical objects, especially through "content dictionaries".)

These are very significant advantages that arise simply from including formal material in text. There is, however, a key *fourth element.* Computer systems can act on the formal material to create a large collection of *accessory knowledge,* meta-knowledge, and metadata. For example, given a mathematical theorem, the system can compute all definitions and lemmas on which it depends. This can be collected as part of the accessory knowledge that is hyperlinked to any text containing the formal material.

1.2 Authorship - knowledge programmers and expositors

There is a rough division of purposes in developing mathematical texts. One purpose is to be precise, thorough and correct, while the other is to make explanations that are easily grasped both for their internal content and in relation to other cognitively relevant concepts and applications. Ordinarily an author would have both purposes, but may compromise one to emphasize the other for intended readers.

Let us designate the roles emphasizing these purposes by *expositor* and *knowledge programmer*. We adopt the latter term in reference to those who develop mathematical texts to extreme precision and thoroughness, necessarily with automated assistance of various kinds, because of the similarity in specialized skills and methods between such authors and computer programmers. Let us reserve the term *formal* for the kinds of texts produced by knowledge programmers, in contrast to expository texts.

Mathematical expositors and knowledge programmers have different goals, but have a profound intellectual commonality, namely the appreciation of precise expression and understanding in mathematical terms. They could support one another by coordinating their respective texts to complement their goals. We believe, for reasons to be explained below (Section 1.6), that such formal text can be of great value to expositors generally when they

[1] http://www.openmath.org

incorporate reference to formalized texts into their expository texts. And reciprocally, the exposition of formalized text is essential to making the product of knowledge programmers of value to a wider readership.

1.3 Semantically Anchored Text

Here we exemplify semantically anchored text, then indicate the nature of related formal text, define explicitly what we mean by *anchored text*, and clarify the extent of semantic representation. We start with an example of a fragment of semantically anchored mathematical text. It will illustrate the anchor points in detail. Whenever we say "anchored text," we mean semantically anchored. [We intersperse comments on the text in square brackets.]

Example of anchored text Suppose we want to explain why the square root of two, $\sqrt{2}$, is irrational. This fact follows from more general ones, e.g. that the root of any prime is irrational or that the square root of any nonsquare is irrational. One class of proofs is based on the idea that if we assume that $\sqrt{2}$ is rational, say p/q, then we can produce an unbounded descending sequence of natural numbers, say p'/q' with $p' < p$, then $p'' < p'$, etc. But such a sequence of "infinitely" decreasing nonnegative numbers is impossible. Hence $\sqrt{2}$ cannot be rational.

We can prove this result carefully starting from a formal machine-checked proof that if the natural number a is prime, $prime(a)$, then there cannot be natural numbers p and q, with q nonzero, such that $p \cdot p = a \cdot q \cdot q$. Here is the formal symbolic statement as it occurs in our digital library of formal mathematics:

Thm 1 $\forall a : \mathbb{N}.\ prime(a) \Rightarrow \neg(\exists p : \mathbb{N}, q : \mathbb{N}^+.\ p \cdot p = a \cdot q \cdot q)$

[In this library all defined symbols are connected to their defining texts, and when the text is anchored the reader can access such definitions at will; in Thm 1 we might access these concepts: $\forall a : \mathbb{N}$, or $\mathbb{N}$, or $prime(a)$ or $\Rightarrow$ or $\neg$ or $\exists p : \mathbb{N}$, or $\cdot$ and so forth. We find that $\mathbb{N}$ is the type of natural numbers $\{0, 1, 2, \ldots\}$, that $\forall a : \mathbb{N}$ means "for all a in the natural numbers," that $\Rightarrow$ means *implies,* $\neg$ is *not,* and $\exists p : \mathbb{N}$ means we can find a number p in $\mathbb{N}$. Just below we see the less common notation, $(\mathbb{N} \rightarrow Prop)$, which denotes the class of all *propositional functions* over the natural numbers, e.g. properties such as $n > 0$.]

So Thm 1 says exactly this:

> "For all natural numbers a, if a is a prime number, then it is not possible to find a natural number p and a positive integer q such that p times p is equal to a times q times q."

We precisely express the idea that it is impossible to have an infinite decreasing sequence of natural numbers satisfying a propositional function P over $\mathbb{N}$ by proving this fact:

Thm 2 $\forall P : (\mathbb{N} \rightarrow Prop).\ (\forall x : \mathbb{N}.\ P(x) \Rightarrow (\exists x' : \mathbb{N}.\ x' < x\ \&\ P(x'))) \Rightarrow \neg(\exists x : \mathbb{N}.\ P(x))$

So, to show that a prime number a has no rational square root, it is enough to show that if we could find a ratio p/q, expressed with a non-negative numerator, whose square was a, then we could find a smaller such numerator p' of a rational root p'/q' i.e.,

Thm 3 $\forall p : \mathbb{N}, q : \mathbb{N}^+.\ p \cdot p = a \cdot q \cdot q \Rightarrow (\exists p' : \mathbb{N}.\ p' < p\ \&\ (\exists q' : \mathbb{N}^+.\ p' \cdot p' = a \cdot q' \cdot q'))$

To show this, suppose $p \cdot p = a \cdot q \cdot q$; we show that we can rewrite

$$(p \cdot p = a \cdot q \cdot q) \text{ to } (q \cdot q = a \cdot p' \cdot p')$$

and then

$$(q \cdot q = a \cdot p' \cdot p') \text{ to } (p' \cdot p' = a \cdot q' \cdot q')$$

giving us a rational square root with numerator $p' < p$ and denominator q'. These rewrites are justified by the following special-purpose theorem, and they finish the proof that a is irrational:

Thm 4 $\forall a : \mathbb{N}.\ \mathit{prime}(a) \Rightarrow (\forall p : \mathbb{N}, q : \mathbb{N}^+.\ p{\cdot}p = a{\cdot}q{\cdot}q \Rightarrow (\exists p' : \mathbb{N}^+.\ p' < p\,\&\,q{\cdot}q = a{\cdot}p'{\cdot}p'))$

Now imagine that the above text is online and all the mathematical symbols and references are linked to definitions and explanations of the symbols and to the various other entities referred to by the text. What is remarkable about this example is that from a single connection to the formal material — even one definition such as $\mathit{prime}(a)$ — we derive a wealth of benefits. This is because the formal material is richly interlinked. We can ask, for example, to display all theorems which use the concept *prime*. We can list all definitions and lemmas that are used to prove any theorem, and thereby acquire a sense of how *logically deep* the result is. We can also "execute" certain theorems, such as: "it is decidable whether a number is prime."

Further, imagine there were a multitude of such expository texts online with such embedded symbols and references. There would then be a precise computer-checkable criterion for when several texts mention the same concept or entity.

Formal text The complete formalized proof of the lemma cited above, to the effect that integers are even when their squares are, is in Figure 1; it is presented in a top-down goal oriented form, including at each step an instruction for how to complete the subproof or reduce it to one or more subgoals. A remarkable feature of the formal material is that all of the underlying proofs can be read online or printed in the text. Typically, these proofs are read in a top-down progression to any level of detail required for thorough understanding, and when a person consulting such a proof is satisfied as to the truth of a subgoal, the proof below it is ignored. We expect that adequate expository text would usually obviate the need for readers to go to the detailed computer checked proof, but this detail is available when desired, and research [24] supports the hope that automated assistance for verbalizing formal proofs will become practical.

A particular value of formalized proofs is that, even if one ignores their value as detailed articulated arguments, one can automatically identify all definitions and theorems cited by

$\vdash \forall a{:}\mathbb{Z}.\ \text{Even}(a{\cdot}a) \Rightarrow \text{Even}(a)$ by Auto

1. $a : \mathbb{Z}$
2. $\text{Even}(a{\cdot}a)$

$\vdash \text{Even}(a)$ by BackThru: Thm* $\forall x{:}\mathbb{Z}.\ \neg\text{Odd}(x) \Rightarrow \text{Even}(x)$

$\vdash \neg\text{Odd}(a)$ by Analyze

3. $\text{Odd}(a)$

$\vdash$ False by New:n Analyze3

3. $n : \mathbb{Z}$
4. $a = 2{\cdot}n{+}1$

$\vdash$False by $a{\cdot}a = 4{\cdot}n{\cdot}n{+}4{\cdot}n{+}1$ Asserted

\
. . . . .assertion . . . . .

$\vdash a{\cdot}a = 4{\cdot}n{\cdot}n{+}4{\cdot}n{+}1$ by Rewrite by Hyp:4

$\vdash (2{\cdot}n{+}1){\cdot}(2{\cdot}n{+}1) = 4{\cdot}n{\cdot}n{+}4{\cdot}n{+}1$ by Auto

---

5. $a{\cdot}a = 4{\cdot}n{\cdot}n{+}4{\cdot}n{+}1$

$\vdash$ False by $\text{Odd}(a{\cdot}a)$ By Witness: $2{\cdot}n{\cdot}n{+}2{\cdot}n$

6. $\text{Odd}(a{\cdot}a)$

$\vdash$ False by BackThru: Thm* $\forall x{:}\mathbb{Z}.\ \neg(\text{Odd}(x)\ \&\ \text{Even}(x))$ Using:$[a{\cdot}a]$

Figure 1: A Formalized Proof

them; thus each formalized proof as a whole can be regarded as a single inference to its conclusion from specific definitions and prior theorems, which could be listed on demand from an anchored text. For examples of formalized proofs online see www.nuprl.org/Nuprl4.2/Libraries.

Definition of anchored text This text is *anchored* in the sense that elements of it are hyperlinked to formal terms that can be manipulated by computer systems. We do not presume that the text is *interactive* in the sense that the content of the text is altered by user actions. But clearly some forms of interactivity which assist understanding by exploiting the formal syntax would be well worth providing; examples include requesting a change of display forms and normalizing certain expressions.

What anchored text allows is access to information that the knowledge programmers create by interacting with the text. They can use the systems to provide a rich collection of

accessory material that is hyperlinked. Some authors of expository documents may feel comfortable using the editing features of the provers to create more accessory content, however, such content has already been created by the knowledge programming community and we expect that in addition to whatever further content that community will produce anyway, knowledge programmers will be attracted to directing some of their efforts in reaction to expositors' interests.

In addition, associated with the formal material is a wealth of *formal metadata.* We can describe the theories in which the results lie, e.g. number theory, list theory, etc., and list all the axioms and primitive concepts required.

Semantic representation Strictly speaking, the semantics of a text is related to its interpretation, the meanings beyond its structure as data. Only computational semantics can be directly implemented on computers, computational methods for referring to values and entities being most tractable. But clearly the semantics intended for interpreting mathematical texts goes far beyond mere stipulation of computation; this is even plainer for informal texts generally. Yet, increasingly, non-computational semantics is in large parts computationally approximated or represented in computer implementations.

Formalized mathematical statements and arguments provide the best examples to date of semantically based practices whose computationally accessible parts have been revealed and implemented. Truth of even precisely formulated mathematical statements cannot normally be specified by an algorithm, and even computationally implemented proof systems must remain incomplete and underdetermined by the actual semantics used to justify those proof systems. Still, the precise computational representation of propositions and semantically justified methods for reasoning about them provide the nearest approach to computer representation of the meanings of texts.

1.4 Formal Digital Libraries

Our conception of semantically anchored texts presupposes access to digital libraries of formal mathematical artifacts into which texts may be anchored, which provide the bases for common reference and semantic significance. Such libraries must be relied upon for the basic navigational and search utilities needed to locate and present formal artifacts, and, as discussed below, should further accommodate data objects "binding" the expository texts to the formal artifacts in which they are anchored. These "binding" data objects should also be susceptible to the finding aids available through the library; they provide valuable informal "metadata" relating formal data by common occurrence with an expository text.

The illustrative material above uses facts proved in the Nuprl proof development system. Similar accounts could be given in any of the other provers we listed above. Here is a PVS definition of prime number for example.

```
prime?(i: int): bool = (FORALL (j: posnat): j /= 1 AND j /= i IMPLIES
                            NOT divides(j,i)) AND i > 1
```

In the course of its verification work, the knowledge programming community generates a large amount of formal mathematics that is of general interest, e.g. theorems about graphs, sets, functions, automata (finite and infinite), formal languages, and concurrent process, as well as about data types such as trees, lists, streams, integers, arrays, and so forth. Further, large amounts of accessory information and formal metadata are created. Much of this material is of interest in mathematics and computer science education.

The algorithmic activity of creating and displaying samples of formal mathematics integrated with plain text has been accomplished using tools of the knowledge programmers.[2] But the *social success* of this activity will require dissemination to a broad audience and integration into on-line libraries, as well as *tools* for people who are not trained in formal mathematics to use the formal material in creating new documents.

1.5 Authoring Tools

We know how to produce semantically anchored mathematical documents using formal material from a Formal Digital Library. We want to create simpler means for non-experts to readily produce such material. The idea is to enhance editors for writing mathematics, using enhanced editors that can also access formal material.

We imagine the current situation in which knowledge programmers use provers to create formal material. Expositors create semantically anchored articles using this material, and they might also pose questions or challenges to the knowledge programmers. For every expert, we imagine that a hundred or more expositors will use the formal mathematics in documents because they will know that it is correct, and they will gain computer-mediated connectivity to hundreds of related results by adding this material to their documents.

Basic parameters of the authoring problem are: the form and media in which the anchored texts and formal texts exist; the editor used by the expositor for creating anchored texts; the kind of access to formal texts to be provided to the author; and the methods provided for rendering formal text within expository text.

For example, the formal texts might be stored as web pages or in a repository accessed by some specific API. Expository texts might be created as Latex sources, HTML sources, or in a format for a word processor, perhaps incorporating MathML.[3] The kind of connections between the expository texts and formal texts in which they are anchored may vary according to their respective media. The most intimate connection between formal and expository texts would be coexistence in a repository of texts with a common formalization of reference, and to which common rendering tools and other utilities could be applied.

A minimal useful mediation between an expository text and the formal material in which it is anchored would be a file, such as a plaintext or pdf file, stored with a data object referring to the anchoring formal texts, along with informal instructions for discerning the anchored points in the expository text and associating them with their formal anchors. In this case

[2]See http://www.cs.cornell.edu/Info/People/sfa/Nuprl/HanoiTowers/Xhanoi_basic_solution.html for an example of anchored text online.

[3]Examples of commercial text processing systems are WebEq and Integre's mathematical text editor. See http://www.dessci.com/en/products/webeq and http://www.integretechpub.com/.

the expository text component can be rendered independently of the formal text, and is essentially treated as an unstructured "blob" (binary large object) by computer processes designed to exploit the connections to formal texts. To these processes, the expository text is treated simply as the data object associated with it in which the machine-recognizable anchoring actually occurs.

A workable intermediate relation between expository texts and formal anchor texts would be, like the minimal meditation above, to pair the author's expository text source[4] with a data object directly referring to the formal anchor texts, but instead of *informal* instructions for discerning the anchor points in the expository text source, their is a machine recognizable criterion for recognizing them. And instead of the informal source being independently rendered, programs determine how to render mathematical text occurring in the expository text based upon the formal anchor text at each anchor point.[5]

1.6 Value of Semantically Anchored Mathematical Text

Basis for a semantic web of mathematics Some of the advantages of semantically anchored mathematical texts are clear from the material on the irrationality of roots of primes. The definitions and theorems are completely rigorous, and they are linked by hypertext, including links to informal explanations of the primitive notions and axioms. As described below, all the details of proof are accessible and proofs can be read top down to progressively finer levels of detail.

What is less visible is the underlying "semantic web" created by saving all logical dependencies as explicit links. This is an extraordinary resource that can be used in semantic search and computer-aided semantic processing.[7]

Machine assisted production methods will enable the capture of a large amount of formal material, making it available for multiple purposes in many articles and documents. It could be that some material will be directly included in thousands of documents — imagine how many textbooks and articles give definitions of prime number or of a graph (usually different in small details).

These advances will also make possible a dynamic of the kind seen in chat rooms and online forums, where the knowledge programmers inject a formal result into a dialogue or lesson.

In addition to allowing authors to refer directly to formal content, anchoring will also enable a new methodology for collecting and improving pre-existing material. As quality ordinary texts are assembled, they may attract annotators who will assign semantic anchors; among those annotators may be knowledge programmers who would further develop suitable formal material in response. This makes the documents more rigorous, full and solid.

[4]Latex is widely used by mathematical authors, and so is among the target formats that are likely to repay the effort of enhancement.

[5]One of the intended uses of MathML is to build utilities that generate "presentation" code from "content" code.[6]

[7]It is also the basis for interactive experiments of the kind that can be seen even in the Web account of the sample lessons at http://www.cs.cornell.edu/Info/People/sfa/Nuprl/eduprl/Xcounting_intro.html.

It "hardens" them as reference resources. It also incorporates them into a semantic mathematics Web for a large collection of formally supported documents. This collection will in turn make it easier to create more formal material, igniting a self-sustaining feedback cycle.

Nucleating large scale collaboration A significant advantage of using formal content in mathematical text is that the body of formal material is known to be completely coherent in *every detail*. The definitions work exactly with the theorems that use them. The examples all reference the same exact definitions directly from instance to instance, or else they may reference variant definitions that are formally related in the repository; the functions have exactly the domain they need.

This is not true of purely informal texts. There a theorem might use one definition in its statement and a slightly different, perhaps equivalent, one in some step of a proof or in a lemma, without resolving the difference. This is one reason that so few textbooks or articles rely on others. For example, three books about finite automata might use three slightly different definitions of finite automata. They can't even share examples exactly. Two different accounts of factorization might use different definitions of prime number — e.g., one in terms of abstract algebra using "associates," the other using the concept of "no proper divisors."

Writers who incorporate formal definitions will know that all the definitions, theorems, and proofs work together. They may appreciate the enormous amount of work that goes into achieving this interoperability. They might know that this formal material is like diamonds — very strong, solid and clear. The impact of a cooperative group of writers and readers using precisely interoperating concepts will be quite extraordinary.

Educational value The semantically anchored documents and means of producing them will have educational value as well. First, they produce what we call *formally-grounded explanations,* that is, an explanation that can be reduced to readable machine-checked proofs in a formal logic. These have educational value because they overcome known problems with traditional mathematical texts [1] and documented difficulties teaching mathematical problem solving [20]. It will also reinforce ties between mathematics and computing. We treat these topics in Section 2.1. Access to proof has educational merit on its own [18, 17, 42].

Enhancing formal texts The benefits of anchored texts mentioned above have largely been in contrast to purely informal expositions. But another value of anchored expository texts is in contrast to purely formal texts. The near futility of trying to understand large collections of unfamiliar formal texts with no informal expository context has surely been experienced by most who have attempted to use such collections. The paucity of such informal expositions linked to formal materials is a serious obstacle to widespread exploitation of such resources, and lowering the technical barriers to exposition may help. Expository texts significantly amplify the formal work. Not only is the formal work given a context of significance that may not inhere in the formal aspect itself, but the knowledge programmer

is likely to be influenced in formal development towards organizations of material that can be exposited and further contribute to other expositions.

2 Semantically Anchored Educational Material

2.1 Pedagogical Issues

The semantically anchored texts that we produce contain *formally-grounded explanations;* by this we mean an explanation that ultimately can be reduced to a readable machine checked proof in a formal logic. Notice that because explanation is so fundamental to education, every potential advantage in teaching it is heavily weighted. Whether we are talking about an English essay, a chemistry experiment, a legal argument or a mathematical demonstration, college students are taught to answer the question "How do you know?" They are taught to give evidence and to say what statements follow from others. This ability to provide evidence and evaluate arguments is critical to a liberal arts education or an engineering one. What is the educational value of formally-grounded explanation, and what known pedagogical problems does it solve? It solves known problems with traditional mathematical texts, and it helps overcome documented difficulties teaching mathematical problem solving. It also reinforces ties between mathematics and computing that are known to help teach the basic concepts of function, induction, and proof that are fundamental, and yet problematic to teach.

Consider the issue of standard mathematics textbooks. Some do not bear logical scrutiny, at best they contain small but annoying errors, at worst they contain major conceptual errors. Even the best of them suffer from errors and omissions of detail that students waste countless hours puzzling over. Reference books and textbooks alike exhibit the problem of locating key information, such as definitions, notations or theorems. Good texts have large indexes, but even they are tedious to use and often fail. The problem of missing motivation to proof steps leaves readers to wonder why a simpler justification they have in mind is inadequate. The courseware that can be produced from anchored texts alleviates these problems.

The underlying formal proofs themselves also have value for learning how proofs work. Studies [1] have shown that students have difficulty understanding the goal and subgoal structure in solutions to problems, especially those solutions given by induction. This is also clear from books devoted entirely to proof [11, 45, 40] and studies in logic [37, 32, 17]. The formal proof structure we adopt is excellent for alleviating this difficulty.

Our existing formal reference material is especially suited to relating computational and mathematical concepts; relating functional programs to mathematical functions is known to help with the problem of teaching functions [20, 1]. We know from direct experience and from the literature [41, 17, 18] that some of the fundamental concepts in mathematics, such as *function, induction* and *proof* are difficult to teach. These concepts are central in both continuous and discrete mathematics, and the function concept seems to be critical in understanding *abstraction.* These concepts are basic to the language of modern science and engineering. They are as important in computer science as they are in physics.

2.2 Educational Opportunities

The existence of large scale collections of semantically anchored texts can be useful in teaching *facts, techniques* and certain modes of *understanding*. Here is how.

1. The level of detail at which students interact with the material is *flexible*, from high level summaries and guided readings down to an examination of *every detail* and every lemma required for a proof.

   Often students are frustrated because a step in an argument that was trivial to the author is not clear to them or because some critical fact was left implicit. Since these proofs are complete, all the details are present, but on demand.

2. Providing multiple anchored texts that use the same definitions and proofs allows the instructional staff to offer many different approaches to the same mathematical ideas. So there are *diverse entry points* to match a diverse student body.

3. The refinement style for proofs helps teach *problem solving technique*. The vocabulary of *goals, subgoals, rules,* and *lemmas* enables teachers to discretize and quantify some of the learning [38].

4. Since formal language can express virtually any mathematical concept, we can begin to *teach understanding* as a process of making connections between topics among the large coherent body of material. As research libraries are made accessible through expository articles, we will be able to relate such ideas as induction and recursion, algebraic structures and program modules, graphs and relations, across many accounts.

3 Efficiencies from Anchoring in Formal Materials

Here we focus on efficiencies, some of which have been indicated above, that may be realized by exploiting formalized mathematical resources.

Establishing Coherency When an author marshals a collection of concepts, entities and facts for presentation, there are issues of coherency between them. For example, suppose an expositor aims to explain the point and correctness of some algorithm. This explanation may depend on deploying a variety of pertinent concepts and facts. Do the formulations and definitions on hand work together well? Are the authoritative texts the expositor cites really using precisely the same concepts? When material is completely informal these issues must be investigated and established by the author and ultimately by the readers as well if they are to know what the author knows.

A library of such concepts and entities would obviate that effort of ascertaining intended common reference and all it entails. Note that it is the formality (computer-recognizability) of reference that matters to this point, and not the formality (computer-checkability) of correctness of facts. If the commonly referenced library is considered authoritative then one

may explore the space of established results about common concepts and entities, and use the existence of a large authoritative body of texts based upon them as evidence for coherency and fruitfulness.

Even if *not all* proofs are formalized, the fact that a formulation has been exercised by *some* formalized argument can prevent some errors in formulation of concepts. And within informal arguments, the use of precise claims and explicit references to facts upon which the informal inferences are based has obvious value when readers must ascertain the value of such informal arguments.

Accessibility of Detail and Verification of Correctness If in addition one includes in the library computer checked proofs of facts, then such proofs make two sorts of further contribution to the economy of sharing formalized concepts and entities.

One contribution is that such proofs effectively provide precise explanations of reasoning that can be probed if need be for extreme detail. Because of the nature of practical computer checkable proof, this resource may be expected to have limited appeal for many readers of expositions, although it could be of more value to the expositor. Yet, even when one ignores the internal structure of a formal proof one can still ascertain the definitions and prior theorems upon which the proof's claim is based.

The other contribution is as a relatively inexpensive alternative to the expensive social process of many persons scrutinizing an informal argument for errors. Consequently, even if the detailed automated proofs were not used directly as resources by readers or even expositors, the fact that proofs are computer checked can be expected to make a greater amount and variety of coherent and reliable material available without having to wait for the social process of looking for errors in the proofs to catch up. (One must still get agreement on the reliability of the proof checkers, but that doesn't have to be re-established anew every time a new proof is added.)

Thus, reliable innovations can be more rapidly deployed and be adopted for informal expositions.

Division of Labor There will probably always be far more potential expositors than there will be knowledge programmers because of the degree of specialized training involved. Even if, as seems unlikely, most knowledge programmers turned out to be good expositors, there would never be enough of them to satisfy the demand for expositions for various audiences and occasions. In the world we hope for, the population sizes for participants would probably stand in the relation:

$$\#\text{knowledge programmers} \ll \#\text{expositors} \lll \#\text{readers}$$

It would be a waste to condition the authorship of the kinds of expository texts we have in mind upon an expositor's mastery of knowledge programming; consequently, we must find practical methods of cooperation between expositors and knowledge programmers. Of course, there is a broad basis of common understanding possible between expositors and knowledge

programmers – the processes of formulation, definition, and argument are understood by both parties.

As a library of concepts, accumulated facts, and expositions grows, we expect significant reuse of formalized material by multiple expositions. Naturally, an expositor will sometimes wish to have a knowledge programmer develop new material for an exposition. This would lead to the development of a graduated series of utilities whereby the expositor would be able to direct more of the knowledge programming personally rather than negotiating the whole development with a knowledge programmer. (Analogously, the relations between computer programmers and their clients can vary in how much programming the clients are able and willing to do themselves and how much they get the programmers to do. Similarly, the relations between an author and a clerical assistant may vary according to how much word processing the author chooses to do.)

References

[1] H. Abelson and G. J. Sussman. *Structure and Interpretation of Computer Programs*. MIT Press, Cambridge, MA, 1985.

[2] A. Asperti, L. Padovani, C. Sacerdoti Coen, and I. Schena. HELM and the semantic math-web. In Paul B. Jackson and Richard J. Boulton, editors, *Proceedings of the 14th International Conference on Theorem Proving in Higher Order Logics (TPHOLs'01)*, volume 2152 of *LNCS*, pages 59–74. Springer, 2001.

[3] James Caldwell. Intuitionistic tableau extracted. In *International Conference on Analytic Tableaux and Related Methods*, 1999.

[4] O. Caprotti, D.P. Carlisle, and A.M. Cohen. The OpenMath Standard `http://www.openmath.org/cocoon/openmath/standard` The Open Math Society, `http://www.openmath.org`, 2002.

[5] David Carlisle, Patric Ion, Robert Miner, and Nico Poppelier. Mathematical Markup Language (MathML) version 2.0. W3C Recommendation, World Wide Web Consortium, 2001. Available at `http://www.w3.org/TR/MathML2`.

[6] Sudarshan S. Chawathe, Anand Rajaraman, Hector Garcia-Molina, and Jennifer Widom. Change detection in hierarchically structured information. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 493–504, 1996.

[7] Robert L. Constable. Creating and evaluating interactive formal courseware for mathematics and computing. In Magdy F. Iskander, Mario J. Gonzalez, Gerald L. Engel, Craig K. Rushforth, Mark A. Yoder, Richard W. Grow, and Carl H. Durney, editors, *Frontiers in Education*, Salt Lake City, Utah, November 1996. IEEE.

[8] Robert L. Constable, Stuart F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, Douglas J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, James T. Sasaki, and Scott F. Smith. *Implementing Mathematics with the Nuprl Development System*. Prentice-Hall, NJ, 1986.

[9] Cristina Cornes, Judicaël Courant, Jean-Christophe Filliâtre, Gérard Huet, Pascal Manoury, Christine Paulin-Mohring, César Muñoz, Chetan Murthy, Catherine Parent, Amokrane Saïbi, and Benjamin Werner. The Coq Proof Assistant reference manual. Technical report, INRIA, 1995.

[10] Y. Coscoy, G. Kahn, and L. Théry. Extracting text from proofs. In *Typed Lambda Calculus and its Applications*, volume 902 of *Lecture Notes in Computer Science*, pages 109–123, 1995.

[11] Antonella Cupillari. *The Nuts and Bolts of Proofs*. Wadsworth Publishing Co., Belmont, California, 1989.

[12] J. H. Davenport, Y. Siret, and E. Tournier. *Computer Algebra — Systems and Algorithms for Algebraic Computation*. Academic Press, London, 1988.

[13] Andreas Franke and Michael Kohlhase. MathWeb, an agent-based communication layer for distributed automated theorem proving. In Ganzinger [15].

[14] Andreas Franke and Michael Kohlhase. System description: MBase, an open mathematical knowledge base. In David McAllester, editor, *Automated Deduction — CADE-17*, number 1831 in Lecture Notes in Artificial Intelligence, pages 455–459. Springer Verlag, 2000.

[15] Harald Ganzinger, editor. *Proceedings of the 16th International Conference on Automated Deduction*, volume 1632 of *Lecture Notes in Artificial Intelligence*, Berlin, July 7–10 1999. Trento, Italy.

[16] Michael Gordon and Tom Melham. *Introduction to HOL: A Theorem Proving Environment for Higher-Oder Logic*. Cambridge University Press, Cambridge, 1993.

[17] Gila Hanna. Proofs that prove and proofs that explain. In G. Vergnaud, J. Rogalski, and M. Artigue, editors, *Proceedings of the International Group for the Psychology of Mathematics Education*, volume II, pages 45–51, Paris, 1989.

[18] Gila Hanna. Challenges to the importance of proof. In *For the Learning of mathematics*, volume 15. FLM Publishing Association, Vancouver, British Columbia, Canada, November 1995.

[19] Gila Hanna and E. Barbeau. What is proof? In B. Baigrie, editor, *History of Modern Science and Mathematics (4 volumes)*, volume 1, pages 36–48. Charles Scribner's Sons, New York, 2002.

[20] Guershon Harel and Ed Dubinsky, editors. *The Concept of Function, Aspects of Epistemology and Pedagogy*, volume 25. Mathematical Association of America, 1992.

[21] John Harrison. HOLLight: A tutorial introduction. In *Formal Methods in Computer-Aided Design (FMCAD'96)*, volume 1166 of *Lecture Notes in Computer Science*, pages 265–269. Springer, 1996.

[22] HELM: An hypertextual electronic library of mathematics. Home page `http://helm.cs.unibo.it`.

[23] Jason Hickey, Aleksey Nogin, Robert L. Constable, Brian E. Aydemir, Eli Barzilay, Yegor Bryukhov, Richard Eaton, Adam Granicz, Alexei Kopylov, Christoph Kreitz, Vladimir N. Krupski, Lori Lorigo, Stephan Schmitt, Carl Witty, and Xin Yu. MetaPRL — a modular logical environment. *Proceedings of the 16th International Conference on Theorem Proving in Higher Order Logics* (TPHOLs), D. Basin and B. Wolff (eds.), LNCS 2758, pp. 287-303, Springer-Verlag, 2003.

[24] Amanda M. Holland-Minkley, Regina Barzilay, and Robert L. Constable Verbalization of High-Level Formal Proofs In *Proceedings of the Sixteenth National Conference on Artificial Intellegence*, Menlo Park, CA, AAAI Press

[25] Matt Kaufmann, Panagiotis Manolios, and J Strother Moore. *Computer-Aided Reasoning: An Approach*, volume 3 of *Advances in Formal Methods*. Kluwer Academic Publishers, Boston, 2000.

[26] Matt Kaufmann and J. Moore. An industrial strength theorem prover for a logic based on Common Lisp. *IEEE Transactions on Software Engineering*, 23(4):203–213, April 1997.

[27] Matt Kaufmann and J Strother Moore. ACL2 home page. `http://www.cs.utexas.edu/users/moore/acl2/`.

[28] Michael Kohlhase. OMDoc: An open markup format for mathematical docuemnts. Seki Report SR-00-02, Fachbereich Informatick, Universitat des Saarlandes, 2000. `http://www.mathweb.org/omdoc`.

[29] Michael Kohlhase. OMDoc: Towards an internet standard for the administration, distribution, and teaching of mathematical knowledge. In John A. Campbell and Eugenio Roanes-Lozano, editors, *Proceedings of the International Conference on Artificial Intelligence and Symbolic Computation (AISC 2000), Madrid, Spain, July 17–19, 2000*, volume 1930 of *Lecture Notes in Artificial Intelligence*, pages 32–52. Springer, May 2001.

[30] Michael Kohlhase and Andreas Franke. MBase: Representing knowledge and context for the integration of mathematical software systems. *Journal of Symbolic Computation; Special Issue on the Integration of Computer Algebra and Deduction Systems*, 32(4):365–402, 2001.

[31] Michael Kohlhase, Paul Libbrecht, Andreas Franke, George Goguadze, Olga Caprotti, Alberto González Palomo, Manfred Riem, and Arjeh Cohen. OMDoc home page. `http://www.mathweb.org/omdoc`.

[32] I. Lakatos. *Proofs and Refutations: The Logic of Mathematical Discovery*. Cambridge University Press, Cambridge, MA, 1976.

[33] Daniel W. Lozier. Toward a revised NBS handbook of mathematical functions. Interagency Report NISTIR 6072, U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, Gaithersburg, MD, September 1997.

[34] Lawrence C. Paulson. *Isabelle: A Generic Theorem Prover*, volume 828 of *Lecture Notes in Computer Science*. Springer-Verlag, New York, 1994.

[35] Frank Pfenning and Carsten Schürmann. System description: Twelf — a meta-logical framework for deductive systems. In Ganzinger [15], pages 202–206.

[36] The QED manifesto. Internet Report `http://www.rbjones.com/rbjpub/logic/quedres00.htm`, 1995.

[37] R. Scheines and W. Sieg. An experimental comparison of alternative proof construction environments. Department of Philosophy CMU-PHIL-40, Carnegie Mellon University, Pittsburgh, Pennsylvania, August 1993.

[38] Alan H. Schoenfeld. *Mathematical Problem Solving*. Academic Press, Inc., New York, 1985.

[39] N. Shankar, S. Owre, J. M. Rushby, and D. W. J. Stringer-Calvert. *PVS Prover Guide*. Computer Science Laboratory, SRI International, Menlo Park, CA, September 1999.

[40] Daniel Solow. *How to Read and Do Proofs: An Introduction to Mathematical Thought Process*. John Wiley & Sons, New York, 1982.

[41] David Tall, editor. *Advanced Mathematicial Thinking*. Kluwer Academic Publishers, 1991.

[42] William P. Thurston. On proof and progress in mathematics. *Bulletin of the American Mathematical Society*, 30(2):161–177, 1994.

[43] A. Trybulec. On a system of computer-aided instruction of logic. *Bulletin of the Section of Logic PAS*, 12(4), 1984.

[44] Wojciech A. Trybulec et al. Mizar home page. `http://www.mizar.org/`.

[45] Daniel J. Valleman. *How to Prove It: A Structured Approach*. Cambridge University Press, New York, 1994.

[46] WebEQ: Dynamic math on the web. Home page http://www.dessci.com/en/products/webeq.

[47] S. Weibel, J. Kunze, C. Lagoze, and M. Wolf. Dublin core metadata element set, version 1.1: Reference description. DCMI Recommendation, 1999. http://dublincore.org/documents/1999/07/02/dces.

[48] S. Wolfram. *Mathematica: A System for Doing Mathematics by Computer*. Addison-Wesley, 1988.