Technical Aspects of the Digital Library of Mathematical Functions
Dreams & Realities

Bruce R. Miller * Abdou Youssef†

August 31, 2001

Abstract: We describe dreams of what the Digital Librarary of Mathematical Functions (a successor to the *Handbook of Mathematical Functions*) should be and the realities that constrain those dreams. We propose practical initial solutions, in order to ease the authoring of adaptable content: a LaTeX class which encourages a modestly semantic markup style; and a mathematical search engine that adapts a text search engine to the task.

1 Introduction

Dan Lozier has presented an overview of the Digital Library of Mathematical Functions (DLMF) project to revise and extend Abramowitz and Stegun's Handbook of Mathematical Functions[1]. This report addresses some of the technical aspects of the project. The route that we will take is to first describe our dreams of what the DLMF might be, or become. That some of these dreams conflict with reality leads to discussion of how we have adapted to those conflicts.

There are two underlying themes to this presentation. The first theme is finding the most practical way of obtaining a variety of information from the project's authors, much of which may seem unusual to them. The second theme is how to make the information available to users through different media and via appropriate search capabilities. We must extract from the author's manuscripts a close approximation to the semantic content, especially the mathematical content, in a form that will allow the broadest, long-term usage. Please note that this project is a work-in-progress; many of the most interesting problems have not yet been completely solved.

*National Institute of Standards and Technology; Gaithersburg, MD 20899, USA; bruce.miller@nist.gov

†Dept. of Computer Science; The George Washington University; Washington, DC 20052; youssef@seas.gwu.edu

2 Dreams & Realities

2.1 Dreams

That the material must be of the highest quality — *authoritative and validated* — should go without saying. We aim to produce a book, a web site with extensive search capabilities and a CD-ROM version. The material must therefore adapt to *various media.*

Clearly, the electronic formats must provide *search capabilities.* And since the DLMF is primarily mathematics with little text, searching formulas is essential. One would like to search for formulas according to keywords or properties of the formula: e.g. 'addition theorem for elliptic functions'. More intriguing would be searches using mathematical patterns. These should be flexible regarding syntax, not restricted to TeX[2], say. And the matching process must understand the basic properties of commutativity, associativity and so forth.

Along with search capabilities, it would be extremely useful to be able to extract *virtual documents.* For example, one might wish to create a page of addition theorems, or a short booklet of differential equations of certain classes. Thus one must not only find pages, but arbitrary document fragments, or equations, and synthesize a document from them. These, along with existing sections or subsections should be viewable in a screen friendly format, or printable.

The DLMF should provide *layers of detail*. Beyond the dry, telegraphic, front facade of the identities, there should be associated with most elements additional metadata. This data would point to references, original sources, or short tracts going into extra detail or more esoterica.

Often one finds a formula not quite in the terms one needs and one would like to *transform* the expressions. It would be exciting, from within the online book, to substitute variables, or rearrange an expression, to acquire exactly the identity required. One may need to re-expand, apply transformations, other identities, ... Transformed or not, users will likely want to copy the formula into their own documents, graphics programs or computer algebra system, without loss of meaning.

For some people, a differential equation tells them all they need to know about a special function. Other, more visually oriented people, will need *interactive graphics* to explore the function in different regions, on different scales. Graphics in 2D and 3D (or more) will be needed. Above all, these graphics must be 'Honest' in the sense of Fateman [3]; they must not display artificial flat planes where clipped, nor smooth over narrow features, nor succumb to other sampling errors.

2.2 Some Successes

Some of these goals are well on their way to success. Dan Lozier has described the project and the strategy for managing it. Given the project's Editorial Board and the authors they have contracted, the quality of the mathematical content is well in hand.

Saunders and Wang [4, 5] have been making significant progress on 3D graphics, with some 'honest' features. We have been exploring 2D interactive honest graphics, as well. But these topics are not relevant at this meeting.

2.3 Realities

Other elements remain difficult, however.

The idea of 'live mathematics' has been explored in teaching materials, where it is totally appropriate. But in our case, the capability to transform a given identity into an arbitrary form of the user's choosing is an open-ended task. Any non-trivial usage would require access to the full power of a computer algebra system. Being part of the U. S. Department of Commerce, NIST's role is to *support* research and commerce, not to replace it. Although presenting limited alternative views of formulas is reasonable, providing completely live on-line manipulations is outside of our mandate.

On the other hand, providing mathematical *reference data* is an ideal role for NIST. The right representation of the mathematics will allow inclusion in user documents, or insertion into a user's computer algebra system (which will make CAS vendors happy). The problem is thus transformed into the problem of extracting a semantic representation of the mathematics, say in OpenMath [6] or Content MathML[7], from what the authors have written.

While a mathematical structure editing application, say with buttons for every conceivable mathematical concept, might be a good solution for the future, it is not an answer for us at present. The chapter authors were chosen for their mathematical expertise. They are experienced mathematicians who already have their preferred authoring tools, their favorite operating system. For the most part, they do not want to be retrained for new tools, nor do we want to retrain them — we want them to focus on the mathematical content.

In fact, most (but not all!) of these authors are comfortable with TEX or LATEX[8]. While LATEX markup is significantly more content oriented than many systems, it still allows for a great many ambiguities. It has been suggested that one should develop a LATEX package providing completely unambiguous mathematics markup. Again, although this is a good goal for future work — and future authors — it would not be workable for the same reasons given above; it would be too unfamiliar and too verbose, and would not get used.

Search, virtual documents and layered information require the support of a variety of metadata. Annotations are needed to provide layered information. An overabundance of references and indexing keywords are called for; some would appear in the regular bibliography or index, but others used solely for layered information or for the search engine. Constructing sensible virtual documents would need some knowledge of the 'role' of sections and equations, e.g. does it represent a definition, or notation. Unless one is willing to impose a rigid organization on all chapters, one must also include metadata to indicate this role.

These items are fairly unusual requests to make of an author, and yet the authors are the best sources. So, while we accept that we will have to augment and rationalize the metadata that they do provide, we must make it as easy as possible for them to include that information. Furthermore, since metadata is generally hidden from view, we must provide feedback to the authors, and also to the editors, for proofreading purposes of what data is present.

Even with extensive indexing metadata, users must be able to search in the mathematical content itself. Search and retrieval technology has been developing for several decades, and has reached a high level of maturity for *text* search [9, 10]. The formulas, equations, and other mathematical constructs present in the DLMF, however, are symbolic and highly structured. Current search systems and technology do not provide the means for formulating math queries and for searching for equations and other mathematical constructs. Many equivalences, such as commutativity and associativity, are simply not recognized.

Finally, there is the issue of targeting the material to different media. Even were it not for our emphasis on semantic markup, we wish to discourage presentation markup. Not only does it obfuscate the content and waste the author's time, it tends to be specific to a certain layout or column width or medium.

2.4 A path to solution

Our general strategy is an evolutionary one. We have developed a DLMF LaTeX class that strays as little as possible from standard LaTeX markup, while encouraging content over presentation. This 'Modestly' semantic markup is described further in Section 3.

Our initial work on the mathematical search problem adapts and augments a conventional text search engine to meet the needs of DLMF. This involves predigesting the mathematics into a more textual form before indexing, defining and implementing a math query language to express equation queries naturally and transforming the search queries into special forms. While perhaps not ideal, the method has had remarkable success so far. The approach is described in more detail in Section 4.

These approaches will allow us to bring many of the dreams to reality in the short term. It also buys us time to continue research and development on better solutions. Focusing the authors on content over presentation and implementation specific details, will allow us to evolve the handling of the material without restarting the project from scratch.

3 Modestly Semantic Markup

Given these constraints, the first technical objective is to modify LaTeX markup just enough to:

- allow the authors to focus on the mathematical content;
- minimize presentation markup, while supporting different formats and media;
- minimize mathematical ambiguities;
- support proofreading with optionally printed representations of all hidden data.

3.1 Metadata

The indexing, labeling and citation mechanisms of standard LaTeX provide a starting point for metadata. The addition of a few simple macros for annotations and original references forms a good basis. However, as feedback to the authors and editors, as well as to other authors who wish to cross-reference, we also provide means to print this extra data without destroying the flow of the main text. An annotated format has been developed which prints the chapter in a single narrow column, with all metadata printed in the second column, aligned to the material it relates to. Alternatively, the data can be printed in an appendix.

3.2 Optional Material

For practical reasons, a printed version of the DLMF must be constrained in size to not much more than 1000 pages. Electronic versions are much less restricted, and thus may contain additional material. Also, material that is most likely to change quickly (e.g. lists of available software) should probably be relegated to the more dynamic electronic media. Thus we provide LaTeX environments that authors use to mark sections of material as being appropriate only for certain media. Again, feedback is needed for proofreading, so such marked material is indicated by bars in the margin with an appropriate label.

3.3 Multiple Media

The book will be printed in a two column format, whereas other media will generally be in a single column — but without a known, fixed width. We therefore strongly prefer that authors *not* break their formula by hand.

We have adopted Michael J. Downes' equation breaking package, `breqn`[11]. It automatically breaks equations across multiple lines depending on the current column width. Although this package is experimental, and the formulas we encounter stress the package, it is handling the task remarkably well. By hiding the invocation of `breqn` inside the standard `equation` environment, the author ideally is not even aware of its presence.

Nevertheless, there are formulas, as well as tables or other material, that the author knows will be too wide to be successfully stuffed into a narrow column. For this, a `onecolumn` environment is defined. In order to support this, and balanced switching between one and two column modes, we have integrated Frank Mittelbach's `multicol` package[12, Ch. 3]. Again, other than the `onecolumn` environment, the author need not be aware of its presence.

3.4 Mathematical Markup

The range of mathematics represented in the DLMF is considerably narrower than the entire field of mathematics. Thus the range of ambiguities we must contend with is considerably smaller. Nevertheless, there are still problems.

Special Functions: Two of the most widespread ambiguities in the field of special functions are the roles of parentheses and superscripts. A parenthesis may indicate either grouping or function application. Superscripts may indicate a power, a function parameter or a derivative. Both of these are largely resolved by declaring all mathematical functions (as LaTeX macros) and incorporating the various sub/superscript parameters into the macro definition. Thus, any remaining superscript can only be a power or derivative. A parenthesis following a function macro probably contains the arguments (and since we know which special function is involved, we know how many arguments to expect), otherwise it is grouping.

This artificial distinction between the function 'parameters' (e.g. the sub- and super-scripts) and 'arguments' (e.g. the parenthesized, or fenced, arguments) is consistent with LaTeX conventions. While the macro `\sin` 'names' the sine function (without arguments), one typically 'names' the Bessel function by writing J_ν (Strictly speaking, the function J of two arguments, ν and z, is curried to a function of z alone).

Consequently, the LaTeX macro we have defined for the Bessel function takes the parameters as arguments. Thus, one refers to the Bessel function itself by

`\BesselJ{\nu}` $\rightarrow J_\nu$

but the application of the function, say in an expression, would be

`\BesselJ{\nu}(z)` $\rightarrow J_\nu(z)$

Since the formatting of sub- and super- and pre-subscripts, tends to yield somewhat messy markup, eliminating that markup has the benefit of easing both the author and parser's tasks. Leaving the arguments separate allows the author freedom to format the argument list in different formats (e.g. different delimiters, or none).

And of course a non-trivial benefit is that we know which J is being referred to.

Fractions and Derivatives: We further discourage various other typical LaTeX idioms that are likely to be ambiguous. Rather than using a `/` for inline fractions, we prefer to complete the set `\frac` and `\tfrac` with `ifrac` (which, admittedly are providing presentation hints, by indicating normal, text or inline fraction).

Likewise, rather than relying on a parser to wonder at the significance of fractions that have non-canceled 'd' in the numerator and denominator in `\frac{df}{dx}`, various derivative macros `\deriv`, `\pderiv` (along with text and inline variants, and optional 'n'), macros for differentials, etc. have been defined.

Superscripts used for derivatives is another ambiguity, but we have not yet developed a markup for this situation. Author preferences for prime, multiple primes and/or parenthesized numerals will need to be respected.

4 Mathematical Search

To identify the major math search issues, it helps to consider the reasons why conventional search systems are inadequate for math search. We recognize three major reasons.

The first is that mathematical content often involve non-alphabetical symbols that are not understood by current search systems, or at least not rightly interpreted. Terms like `Gamma(1/2)`, `P_n(x)`, `x**5`, or `d^2y/dx^2-x=0` are either meaningless or improperly read and processed by current systems. (It goes without saying that typability on a keyboard does not automatically guarantee proper interpretation and processing.)

The second and more challenging reason is that formulas and equations, as well as other mathematical constructs, have rich structures that convey much meaning. Current search engines are not "aware" of those structures, do not capture or index them, and are thus unable to search for information that involve structural relationships and patterns. A query like `sin(x + log x)` is no different to a current search system than `sin x + log x`. Similarly, `x (y + z)` is misinterpreted as `x y + z`, if interpreted at all.

The third and most challenging reason is that the many equivalent ways in which mathematical terms can be expressed, which correspond to synonyms in text search, are often much more complex than textual synonyms, and thus cannot be fully captured in a thesaurus. A summation or a product of two or more terms can be expressed in many equivalent ways due to commutativity and associativity laws. Polynomials can be expressed in many factored and unfactored forms. Trigonometric terms can be easily substituted by other equivalent trigonometric terms. Indeed, it can be argued that a large part of Mathematics is about the different and equivalent ways of expressing a concept or a quantity. Obviously, current search systems are not equipped to recognize those equivalences and take them into account when searching — indeed, the problem is not solvable in general.

Therefore, the major math search issues can be summarized as follows:

- Recognition and proper processing of mathematical symbols in mathematical content and queries.
- Capturing and indexing mathematical structures.
- Providing a math-appropriate query language and user interface that enable users to express their information needs, which often involve math symbols and structures.
- Developing and integrating techniques for taking into account mathematical synonyms and equivalences — at least some of the more common ones such as commutativity and associativity based equivalences.

The next subsection will discuss some of the approaches that are being taken to address those issues.

4.1 Approaches to Math Search

Broadly speaking, two general approaches suggest themselves. The first approach is an evolutionary one, which augments existing text search systems with math-appropriate searching capabilities. The second approach is to create a math search system wholly from scratch to fully capture & index mathematical content.

The augmentation approach leverages the power and maturity of text search engines, and involves less work. Although it will have inherent limitations as to how much mathematical structure it can capture, it will take us a long way towards effective equation-based searching. It is the approach currently adopted by DLMF, and will be discussed in more detail in the next subsection.

The second approach is much more demanding in time and effort. It will require not only the careful use and integration of various computer algebra and symbolic processing techniques, but also the development of novel indexing and search techniques for which research has barely begun. Certainly, parsing techniques for mathematical expressions have been developed for compilers and computer algebra systems [13, 14]. Those techniques can and should be adopted by this approach as well as by the augmentation approach.

The indexing of mathematical structures, on the other hand, is a largely new subject. Some work has been done on searching for integrals[15]. However, to the authors' knowledge, no work has been published on indexing full mathematical structures in such a way that users can search for structures and substructures. By searching for substructures we mean writing a query that has a fragment(s) of an equation, and asking the system to find all documents (or pages) that have the equation which contains the query fragment(s).

As for incorporating mathematical synonyms and equivalences, work on automated theorem proving [16] and symbolic processing in computer algebra can be drawn upon. Nevertheless, sophisticated equivalence detection is still not a mature technology, and is probably too slow for online searching. Furthermore, it is not clear to what extent the user will want the search system to find documents that have rather non-obvious equivalent expressions of what the user is looking for. On the one hand, a search engine that can look for and detect equivalences can be a powerful tool for research and for the advancement of Mathematics and the sciences through the synergistic discovery of similar (or equivalent) patterns in bodies of literature in seemingly unrelated (sub)disciplines. On the other hand, if every search involves the detection of equivalences, the query response time could become intolerable, the number of returned hits could be overwhelming, and the truly relevant documents may be buried deep in the large numbers of returned hits. Therefore, some control should be given to the users as to what type of equivalences and to what extent they want the search system to consider. The technical aspects for providing this control capability is, therefore, another issue to be faced in math search.

4.2 Search Engine Requirements

To meet the imminent search needs of DLMF, we pragmatically opted for the augmentation approach. The first step was to find a text search engine which has a wide range of capabilities and features to make it amenable to augmentation for math search. The second step was to carry out the actual augmentation.

There are a number of search systems, some of which are in the public domain [17, 18, 19, 20, 21] while many others are commercially available. Clearly, what is important is not which specific system to go with, but what features are the most relevant and useful for adaptation and augmentation to math search.

In our experience, we found the following capabilities and features to be particularly useful:

Boolean and proximity operators: All three Boolean operators (AND, OR, and NOT) and many of the proximity operators should be allowed in queries. Examples of proximity operators are word adjacency operators (for phrase search) as well as unidirectional and bidirectional distance operators that specify the distance between the operands in terms of the number of intervening words or intervening paragraphs.

Fields and field-based search: It is desirable to be able to subdivide any document in the database into parts (called fields), and label each part by a field name. Queries can be targeted not only to whole documents but also to specific fields. An important field might be "equation", another might be "theorem", and yet another might be "chapter-title".

Thesaurus support: The thesaurus would allow the math search system designers to define various terms not only in terms of equivalent synonyms, but also as full-fledged queries (i.e., an expression involving Boolean and/or proximity operators). For example, it was mentioned earlier that markup such as `BesselJ` (and `BesselI`, `BesselK`, `BesselH` etc.) helps in semantics and formatting matters. Should a user enter `Bessel` as a keyword, all those markup words (`BesselJ`, `BesselI`, etc.) will not be recognized as matches of `Bessel`. However, if an entry for `Bessel` is defined in the thesaurus, where `Bessel` is made to stand for

```
(Bessel OR BesselJ OR BesselY ...),
```

a thesaurus lookup will cause the keyword `Bessel` to be replaced by its thesaurus entry, and the markup is then guaranteed to be matched.

Note, however, that such a thesaurus cannot handle mathematical equivalences that involve equations.

Use of surrogate files: The underlying concept is to view a document as a pair of files (D, S). File D is the display file that is presented to the user when the document matches a query and is retrieved. The file S (the surrogate file) is what is indexed at the time of creating the database or when adding the document to the database. The key point is that the two files D and S need not be identical. Indeed, the surrogate file S can have:

- Non-displayable metadata, such as (a) authors' notes, (b) descriptive terms and keywords for equations, formulas, graphs and tables, and (c) annotations.
- Modified representations of mathematical equations/symbols/constructs so that those representations are indexable and searchable by text search systems. Roughly speaking, the modified representation corresponds to *content markup* in the parlance of MathML [7] and OpenMath [6], and the original representation corresponds to presentation markup as in MathML, LaTeX, and other word processing systems. Note that the modified representation could be standard markup such a in MathML, but it need not be, depending on the intended use.

The main reason for having a pair of files is that display files do not have all the textual (and thus indexable) contents needed, and, on the other hand, the surrogate files have contents that are not for display.

4.3 Search Augmentation

Our augmentation entails the following steps and modules:

Math query language: Our language is flexible, intuitive, easy-to-use, and efficiently expressive. It is flexible in that it accommodates a variety of "idioms" for operator symbols, e.g., Matlab, C, Fortran, LaTeX, and others. It is intuitive and easy to use in that it draws on what most users know and use in their daily math reading and writing. It is efficiently expressive in that an equation (or fragment thereof) can be expressed in a minimum number of keyboard characters, without resorting to lengthy and elaborate commands and markups. The details of the language will be in the help file of the system, and will be published elsewhere.

Math query parser/translator: This is a front-end layer to parse and translate equation queries to purely textual-and-numeric queries which use Boolean and proximity operators, capturing in the process as much structure and semantics as possible or desirable. For example, a query of the form `1/(Ai^2+Bi^2)` is parsed and translated to

```
frac (1) (Ai super 2 endsuper plus Bi
super 2 endsuper).
```

Contents transformation module: This module transforms the database mathematical content, especially equations, formulas, and mathematical symbols, into a textual form, again capturing as much semantic and structural content as possible. This module and the query parser/translator perform some of the same processing, in that they produce almost identical outputs. However, they still differ in that the parser's input format, that is, the math query language syntax, is different from the transformation module's input format (e.g., the Latex encoding of math equations in documents). For example, the following Latex expression `$x_2+y^{i-j}$` is transformed into

```
x sub 2  endsub plus
         y super i minus j endsuper.
```

Surrogate-files generator: This module generates the surrogate files that contain the transformed math content as well as other metadata. It clearly calls upon the content transformation module to modify the equation representation into an indexable form, as discussed earlier.

Thesaurus: The thesaurus, which will continue to evolve, contains generic math entries, DLMF-specific entries, and markup related entries.

An example of a generic math entry is that of the term *derivative*, which includes among its synonyms the term *differential* or *diff* for short. An example of a DLMF-specific entry is:

```
U:  (U OR VoigtU OR KummerU OR WhitU
     OR ChebyU OR ChebyUs)
```

which tells the system that U should be interpreted as the OR of generic U, the Voigt function U (through its local markup `VoigtU`), the Kummer function U (through its markup `KummerU`), and so on. Note that the latter example illustrates also the use of the thesaurus for capturing internal markup.

Those modules have been designed and implemented, and a working math search system is in place. The system has math search capabilities that far exceed those of text search engines. Still, the system is work in progress. Additional refinements are being made, and new capabilities are being added.

Among those new capabilities are (1) the proper handling of customary names and symbols that are widely used in the area of special functions, and (2) the support of basic equivalences, including commutativity and associativity, among others.

Further into the future, we will explore searching based on new and emerging encodings such as MathML, OpenMath, and XML. The precise direction that will be taken will be determined according to which of the new encoding standards will be adopted and become widely used and technology-supported.

Also, an all math search engine, created from scratch for that purpose, may be developed, if and when the necessary research in indexing of mathematical structures yields the necessary understanding and techniques. Experience gained from the mathematical and scientific user community regarding math search, especially in reference to query languages, modes of search, and what kinds of math information users usually need to search for, will be an extremely valuable guide into building future math search systems.

5 Conclusions

At this stage of the project, we are still very much focused on obtaining the content. Much of the effort up to this point has gone into TEX programming to provide support for a subtly modified level of LATEX markup. The goal is to allow authors to minimize presentation markup and ambiguity, while maximizing content markup. We try to enable the authors to concentrate on the mathematics and develop the content as painlessly as possible.

We are moving into the next phase, focusing on the transformation to XML and Presentation MathML. This latter, using systems such as `tex4ht`[22, Ch. 4] seems within reach, although some refinement is still needed.

It is clear that as we move into further stages, parsing into fully Content MathML or OpenMath, additional refinement and markup of the mathematics will be necessary, even adoption of more verbose, unambiguous markup. But, with the approach outlined we should already be relatively close to that state. As long as *we* understand the author's intent, we will be able to augment the sources with additional markup and declarations to complete the parsing. But while the tools to carry out that parsing are not yet available, applications to *use* such content markup are also not yet widespread, so the presure is not yet intense.

Between the metadata markup and modestly semantic mathematical markup, we already have sufficient data for the augumented text-based search techniques described here. We believe that with this approach powerful search capabilities will be available to DLMF users. Our approaches are rather generic, and should be immediately applicable to other mathematics databases. If these methods prove insufficient, and once we have made further progress in capturing the complete structure of the mathematical data, we will look into true mathematical search techniques.

References

[1] M. Abramowitz and I. A. Stegun, editors. *Handbook of Mathematical Functions*. National Bureau of Standards Applied Mathematics Series No. 55. U.S. Government Printing Office, Washington, DC, 1964.

[2] D. E. Knuth. *The TEXbook*. Addison Wesley, 1986.

[3] R. Fateman. Honest plotting, global extrema, and interval arithmetic. In *Proceedings of the International symposium on Symbolic and algebraic computation*, page 216. ACM, 1992.

[4] Bonita Saunders and Qiming Wang. Interactive 3d visualization of mathematical functions using vrml. Technical Report NISTIR 6289, NIST, 1999.

[5] Bonita Saunders and Qiming Wang. From 2d to 3d: Numerical grid generation and the visualization of complex surfaces. Technical Report NISTIR 6555, NIST, 2000.

[6] The OpenMath Society. *OpenMath*. http://www.openmath.org/.

[7] World Wide Web Consortium. *Mathematical Markup Language (MathML) Version 2.0*, 2001. http://www.w3.org/TR/MathML2/.

[8] L. Lamport. *LATEX: A Document Preparation System*. Addison Wesley, 2nd edition, 1994.

[9] G. Kowalski. *Infromation Retrieval Systems, Theory and Implementation*. Kluwer Academic Publishers, 1997.

[10] G. Salton and M. J. McGill. *Introduction to Modern Information Retireval*. McGraw Hill, New York, 1983.

[11] Michael J. Downes. Breaking equations. *TUGboat*, 18(3):182, Sept 1997.

[12] M. Goossens, F. Mittlebach, and A. Samarin. *The LATEX Companion*. Addison Wesley, 1994.

[13] A. Aho, R. Sethi, and J. Ullman. *Compilers : Principles, Techniques, and Tools*. Addison-Wesley, 1985.

[14] S. Muchnick. *Advanced Compiler Design and Implementation*. Morgan Kaufmann, 1997.

[15] T. H. Einwohner and R. Fateman. Searching techniques for integral tables. In *Proceedings of the International symposium on Symbolic and algebraic computation*, page 216. ACM, 1995. http://torte.cs.berkeley.edu:8010/tilu.

[16] Monroe Newborm and Monty Newborn. *Automated Theorem Proving : Theory and Practice*. Springer Verlag, January 2001.

[17] PLS/AOL. *CPL and PLWeb*. http://www.pls.com/.

[18] The ht://Dig Group. *ht://Dig*. http://www.htdig.org/.

[19] U. Manber, S. Wu, and B. Gopal. *Glimpse and WebGlimpse*. http://glimpse.cs.arizona.edu/.

[20] *IRF*. http://www-nlpir.nist.gov/projects/irf/.

[21] R. Tennant et. al. *SWISH-E*.
http://sunsite.berkeley.edu/SWISH-E/.

[22] M. Goossens and S. Rahtz. *The LaTeX Web Companion*. Addison Wesley, 1999.