Determining Empirical Characteristics of Mathematical Expression Use

Clare M. So and Stephen M. Watt

Ontario Research Centre for Computer Algebra
Department of Computer Science
University of Western Ontario
London Ontario, CANADA N6A 5B7
{clare,watt}@orcca.on.ca

Abstract. Many processes in mathematical computing try to use knowledge of the most desired forms of mathematical expressions. This occurs, for example, in symbolic computation systems, when expressions are simplified, or mathematical document recognition, when formula layout is analyzed. The decision about which forms are the most desired, however, has typically been left to the guess-work or prejudices of a small number of system designers.

This paper observes that, on a domain by domain basis, certain expressions are actually used much more frequently than others. On the hypothesis that actual usage is the best measure of desirability, this papers begins to quantify empirically the use of common expressions in the mathematical literature. We analyze all 20,000 mathematical documents from the mathematical arXiv server from 2000-2004, the period corresponding to the new mathematical subject classification. We report on the process by which these documents are analyzed, through conversion to MathML, and present first empirical results on the most common aspects of mathematical expressions by subject classification. We use the notion of a weighted dictionary to record the relative frequency of subexpressions, and explore how this information may be used for further processes, including deriving common patterns of expressions and probability measures for symbol sequences.

1 Introduction

Most software that deals with symbolic mathematical information have some pre-defined notion of when expressions are well-formed and, of the well-formed expressions, which are the most desirable. Which forms are deemed most desirable is usually decided by the software system designers, though their experience or preference, and hard-coded into the application's logic. This has made symbolic mathematical software more natural to use in some areas than others, depending on the compatibility of the system designer's choices with the user's needs. As we move toward more sophisticated, knowledge-based mathematical software, this methodology becomes increasingly problematic. In this paper we argue that it is important to understand what forms of expressions are deemed

most desirable in the actual practice of mathematics. We believe that empirical knowledge of which forms of expressions are used most often will lead to more effective mathematical software. For example, this information could be used to guide simplification in computer algebra systems, or to provide disambiguation criteria in mathematical document recognition.

Our initial motivation for this work comes from the area of mathematical handwriting recognition. We note that today's acceptable recognition rates for natural language handwriting is achieved with the aid of dictionary-based methods. For example, if the feature analysis of a stroke could yield either `Hdb` or `Hello`, then `Hello` is chosen because it is in the dictionary. At first consideration, such an approach is not suitable for mathematical handwriting recognition for several reasons: Mathematical expressions are trees, not strings. There is no fixed vocabulary from which to build a dictionary. The set of symbols alone is insufficient, and the set of possible expressions is infinite.

Nevertheless, any mathematically sophisticated person can take an arbitrary volume from a mathematical library, leaf through the pages, and, in a few seconds, have a very good idea of the precise mathematical subject area, in part, simply be noticing some characteristics of the formulae. We therefore claim that there is, in fact, usage knowledge that can and should be used by mathematical software packages. In the mathematical handwriting recognition case, this knowledge could be used to disambiguate between $\sin \omega t$ and $\sin wt$, since the former occurs much more often in practice. In the computer algebra case, this knowledge could be used to order one polynomial as x^2+1 and another as $1+\epsilon^2$.

The goals of this present line of work are to understand how
- to capture and represent empirical mathematical usage information
- to employ this information in mathematical software packages
- to analyze and organize this knowledge so as to be most useful.

We report here on our initial results towards these long-term goals. As stated earlier, we see immediate applicability to mathematical handwriting recognition and to symbolic mathematical computing. Other potential applications include mathematical searching, automated classification of mathematical documents, and mathematical data mining.

The contributions of this work are
- the identification of empirical mathematical usage as an important source of information for mathematical software design
- an approach to empirical analysis of mathematical expressions
- specific findings on symbol usage, on a subject-by-subject basis
- specific findings on most common expression usage
- methods to derive pattern expressions, and symbol-sequence Markov chains, based on analysis of instances.

The rest of the paper is organized as follows: We present the methodology of the current study in Section 2. As part of this study, we rely on a TEX to MathML conversion. Section 3 describes this process and extensions we have had to make for the present work. Results on frequency of symbols, as identifiers and operators, are reported in Sections 4 and 5. We present some initial results on expression analysis in Section 6. Section 7 concludes the paper.

#	Subject Classification	#	Subject Classification
19	00 General	34	45 Integral equations
39	01 History and biography	1066	46 Functional analysis
228	03 Math. logic and foundations	543	47 Operator theory
1212	05 Combinatorics	164	49 Calculus of var.; optimization
164	06 Order, lattices, ordered alg. struct.	171	51 Geometry
48	08 General algebraic systems	435	52 Convex and discrete geometry
1383	11 Number theory	1717	53 Differential geometry
108	12 Field theory and polynomials	226	54 General topology
667	13 Commutative rings and algebras	627	55 Algebraic topology
2445	14 Algebraic geometry	1618	57 Manifolds and cell complexes
240	15 Lin. and multilin. alg.; matrix thy	920	58 Global analysis, an. on manifolds
861	16 Associative rings and algebras	877	60 Prob. theory and stoch. processes
760	17 Nonassociative rings and algebras	105	62 Statistics
404	18 Category theory; hom. algebra	209	65 Numerical analysis
239	19 K-theory	237	68 Computer science
1169	20 Group theory and generalizations	113	70 Mechanics of particles and systems
472	22 Topological groups, Lie groups	34	74 Mechanics of deformable solids
185	26 Real functions	69	76 Fluid mechanics
123	28 Measure and integration	13	78 Optics, electromagnetic theory
308	30 Functions of a complex variable	6	80 Classical thermodyn., heat xfer
59	31 Potential theory	553	81 Quantum theory
797	32 Several complex var. & an. spaces	260	82 Stat. mechanics, struct. of matter
312	33 Special functions	48	83 Relativity and gravitational theory
295	34 Ordinary differential equations	6	85 Astronomy and astrophysics
746	35 Partial differential equations	15	86 Geophysics
706	37 Dyn. systems and ergodic theory	96	90 Operations research, math. prog.
52	39 Difference and functional eqns	42	91 Game thy, econ., soc. & behav. sci.
21	40 Sequences, series, summability	35	92 Biology and other natural sciences
88	41 Approximations and expansions	115	93 Systems theory; control
290	42 Fourier analysis	128	94 Info. and comm., circuits
143	43 Abstract harmonic analysis	12	97 Mathematics education
43	44 Integral transforms, op. calculus		

Fig. 1. Count of articles by MR Subject Classification

2 Methodology

To study the empirical usage of mathematical expressions, the first step was to identify a suitable source of mathematical input. A number of possibilities existed, including

- to use logged input from a software system, such as Maple,
- to use a collection of documents from a set of cooperative authors,
- to use the articles from a particular journal

Although any of these avenues would have been easy to follow, each had its own problems: Logged input from a software system would heavily influenced by the characteristics of the system, and thus be riddled with artefacts. Articles from a small set of authors, or from a particular journal, would likely be heavily slanted in their usage and could not be taken as representative.

Instead, we chose to use the collection of articles available on the widely used, public e-Print server, `arXiv.org` [2], as our corpus of mathematical usage. This has the advantage of broad coverage by mathematical area. It also has the disadvantages that:

- Some areas are disproportionately represented.
- The mathematical material is at a research level, and this may not be representative of usage at more elementary levels.
- The material is relatively new, and is not representative of historical usage.

Bearing this in mind, we decided that the collection of articles was sufficiently representative of current mathematical usage to be useful, and that developing a collection that was more balanced by area, level, historical period, *etc*, was a long-term project.

One of the attractive properties of `arXiv.org` is its organization of articles according to the Mathematics Subject Classification, which is used to categorize items covered by the two reviewing databases, Mathematical Reviews (MR) and Zentralblatt MATH (Zbl). The current classification system, MSC 2000 [3], is a revision of the classification scheme that had been used previously by these databases. It consists of more than 5,000 two-, three-, and five-character classfications, corresponding to increasingly finely defined disciplines of mathematics. For example, "11" represents Number theory; "11B" Sequences and sets, and "11B05" Density, gaps, topology.

We followed the following steps to obtain our corpus of expressions to analyze:

The first step was to obtain all articles from `arXiv.org` from the five year period 2000–2004. This data range contained all articles since the new subject classification was introduced. To understand area-specific usage patterns, while having a sufficient number of articles in each category, we grouped articles according to their top-level, two-digit MSC classification. The count by classification of articles considered is shown in Figure 1. Altogether 22,289 articles were accessed. Of these 21,677 came with TEX source. This comprised 4.65GB of PDF files and 794 MB of TEX source.

The second step was to extract mathematical expressions from the articles. It was helpful that the articles had TEX source, but this was not usable directly for our analysis. The problems with TEX source include:

- Mathematical expressions typically use author-defined macros.
- Mathematical expressions my be hidden in macros, and not be visible in the source text.
- TEX expressions typically have only as much structure as is needed to give proper visual grouping. For example `$(ad-bc)^2$` consists of a single row of 7 items, `(`, `a`, `d`, `-`, `b`, `c` and `)^2`. Note that there is no notion that ad and bc are subexpressions, while $d - b$ is not, and note that it is only the closing parenthesis that is squared.

We used our TEX to MathML [1] converter, described in [8], to resolve these difficulties, and performed our analysis on the resulting MathML expressions. The benefit of this approach was that the expressions treated were (for the most part) complete, well formed, and grouped appropriately. The difficulty with the approach was that not all the complexities of TEX were handled, and some expressions were incorrectly translated. However, since we are interested in the most frequently occurring expressions, the incomplete handling of infrequently occurring expressions is not, in principle, a problem. We describe the conversion process in more detail in Section 3. The overall conversion process required about three days of computer time on a personal workstation.

The third step was to examine the MathML expressions for each area, and to build three frequency tables. The first two tables contained counts of all identifier symbols (typically single letter operands) and all operator symbols. The third table counted the number of occurrences in the classification of each sub-expression. These tables were built using syntactic comparison of XML elements. For example, `<mrow><mo>(</mo><mi>a</mi><mo>)</mo></mrow>` would be treated as inequivalent to `<mfenced><mi>a</mi></mfenced>`. We therefore preprocessed the MathML to remove multiple representations for what would appear as *syntactically equivalent* mathematical expressions. This consisted of a number of simple conversions, including

- for `<mi>` and `<mo>`, normalizing the use of the `mathvariant` attribute
- for `<mfrac>`, eliminating any non-zero `linethickness` attribute
- for `<mfenced>`, convert to `<mrow>` with explicit open and close operators
- for `<mmultiscripts>`, convert to `<msub>` and `<msup>`
- elimination of a number of attributes and elements

3 TEX to MathML Conversion

The conversion of TEX to MathML is not a straightforward process. There is not yet a standard tool that completely solves this problem. TEX documents are, in general, programs with the computational power of a Turing machine. In practice, TEX macros are usually used to perform simple substitutions, with a smaller number performing heavy computations and transformations.

There are two principal approaches to TEX to MathML conversion: The first approach is to use alternative style files with modified definitions for the standard mathematical macros. These modified macros leave special markers in the generated `dvi` file, which are then used to generate the MathML. This approach has the advantage that all TEX files can be handled. The disadvantage is that all the high-level structure implicit in the TEX markup is discarded. This is the approach taken by the Hermes project [4].

The second approach is have a (partial) implementation of a TEX processor handle the input, and to generate MathML from the higher-level TEX operators. This has the advantage that implicit semantics in TEX markup (e.g. grouping information from braces,"{" and "}") is available to the MathML generation.

The disadvantage is that, in principle, a complete TEX re-implementation is needed.

For this study, we used a TEX to MathML converter, developed within the ORCCA research group. This converter adopts the second approach. It has a partial implementation of the TEX programming language sufficient to expand the macros of interest in mathematics. Source for a TEX document may be given as a single file, or as a tree of files and using external macro packages. The correspondences between TEX and MathML are given by a set of bi-directional mapping files. These mapping files are intended to allow high-level semantic mappings between TEX and XSLT style sheets [8]. Because complex TEX macros are almost always given in style files, rather than being specified at top-level by authors, the mapping files may almost always be used to eliminate any shortcomings arising from the incomplete implementation of TEX. This translator is available on-line [5].

The conversion of all TEX source documents in the five year `arXiv.org` collection served as heavy test for the MathML converter, and a number of problems were encountered. Initially only 14,354 of the 21,677 articles could be handled automatically. First, we discovered that there were a number of TEX constructs that were not handled by the converter. The most important of these were (1) the handling of explicit positioning commands, e.g. for kerning symbols, and (2) the ability to handle arbitrary external macro packages from a search path. Dealing with these difficulties proved to be fairly easy.

The second major difficulty in the TEX to MathML translation was that a significant number of the TEX source files did not contain valid TEX. The TEX converter had been constructed assuming valid input, the idea being that an author would first produce a correct file by debugging with TEX and then, possibly long afterward, generate MathML. This assumption proved invalid — authors do not always correct their TEX errors if TEX's error recovery gives a desired output. We therefore were required to extend the TEX to MathML converter to simulate TEX error handling.

With user error handling in place, we were able to process 19,137 of the articles automatically. Of these, 19,063 were able to have their MathML canonicalized, and it is from these that we have extracted the expressions for analysis.

4 Identifiers

Our first analysis determines the most frequently occurring symbols used as identifiers in mathematical expressions. By this we mean symbols that occur as operands or function names, rather than as operators.

We counted all symbols occurring in expressions and recorded the results both for the global analysis and independently for each category. The first observation is that in each classification some symbols occur much more frequently than others, and which symbols are the most frequent differs from classification to classification.

All		
Ucode	**Id.**	**Freq.**
006E	n	23,419
0069	i	21,065
0078	x	17,671
006B	k	15,602
0074	t	12,639
0058	X	11,348
006A	j	11,213
0070	p	11,110
0041	A	11,058
0061	a	10,425
0064	d	9,470
006D	m	9,363
0066	f	8,863
004D	M	8,819
0073	s	8,583
0072	r	8,393
0043	C	8,230
0053	S	8,019

03		
Ucode	**Id.**	**Freq.**
0069	i	26,055
006E	n	24,372
0078	x	20,740
0058	X	17,111
0041	A	15,080
0070	p	13,285
03B1	α	12,432
006B	k	12,316
0066	f	11,455
0061	a	11,133
0047	G	11,108
006D	m	10,054
006A	j	9,125
03C9	ω	9,102
004D	M	8,719
0053	S	8,651
0043	C	8,643
0046	F	8,474

11		
Ucode	**Id.**	**Freq.**
006E	n	27,743
0070	p	19,216
006B	k	18,228
0078	x	16,828
0069	i	16,735
0061	a	12,064
006D	m	11,272
0064	d	10,634
0071	q	10,393
0073	s	10,164
006A	j	10,086
0072	r	9,391
0074	t	9,371
0047	G	9,355
0058	X	9,314
0041	A	9,110
004B	K	9,014
0066	f	8,643

35		
Ucode	**Id.**	**Freq.**
0078	x	24,096
0074	t	23,206
0075	u	18,543
006E	n	16,618
006B	k	13,927
0069	i	13,469
0073	s	11,744
006A	j	11,620
0064	d	11,214
004C	L	9,818
03B5	ϵ	9,653
03BB	λ	9,396
0070	p	8,892
0043	C	8,121
03B1	α	7,952
0072	r	7,835
0076	v	7,827
0061	a	7,414

Fig. 2. The most frequent identifiers (per million) in all classifications (All), Logic (03), Number Theory (11) and Partial Differential Equations (35).

03		
Ucode	**Id.**	**Freq.**
03B1	α	12,432
0066	f	11,455
0047	G	11,108
006D	m	10,054
03C9	ω	9,102
004D	M	8,719
0053	S	8,651
0043	C	8,643
0046	F	8,474
0079	y	8,470
0054	T	7,885
0062	b	7,716
004B	K	7,652
0042	B	7,581
0063	c	7,370
0050	P	7,367
0073	s	7,176

11		
Ucode	**Id.**	**Freq.**
006D	m	11,272
0064	d	10,634
0071	q	10,393
0073	s	10,164
0072	r	9,391
0047	G	9,355
004B	K	9,014
0066	f	8,643
0046	F	7,879
004C	L	7,591
004E	N	7,408
0053	S	7,262
0076	v	6,856
0054	T	6,735
0067	g	6,523
0050	P	6,427
007A	z	6,357

35		
Ucode	**Id.**	**Freq.**
0075	u	18,543
0073	s	11,744
0064	d	11,214
004C	L	9,818
03B5	ϵ	9,653
03BB	λ	9,396
0043	C	8,121
03B1	α	7,952
0072	r	7,835
0076	v	7,827
0079	y	7,409
0066	f	7,082
03BE	ξ	7,053
007A	z	6,729
0054	T	6,671
004E	N	6,472
006D	m	6,377

Fig. 3. Most frequent operators (per million) in Logic (03), Number Theory (11) and Partial Differential Equations (35), excluding the 10 most frequent from all categories.

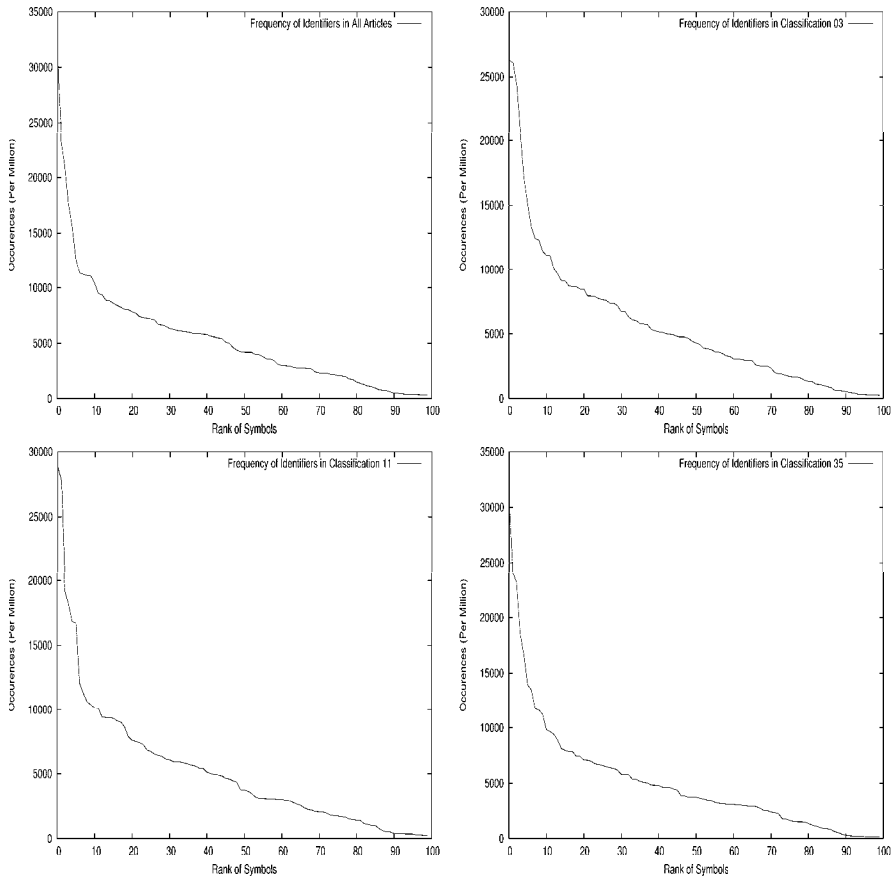

Fig. 4. Most frequent identifiers in all expressions (upper left), Logic (upper right), Number Theory (lower left), and Partial Differential Equations (lower right).
The horizontal axis gives the symbol (from most to least frequent), and the vertical axis gives the number of occurrences per million symbols in the classification.

Figure 2 shows the most frequently occurring identifiers for all the classifications taken together, as well as the most frequently occurring identifiers for three typical classifications, Logic, Number Theory and Partial Differential Equations. For detailed information on all classifications see [7].

This information could be used for disambiguation in mathematical handwriting recognition. In Number Theory, for example, we see that the letter n occurs more than twice as frequently as the letter r. By feature analysis alone, these two letters are difficult to distinguish. This frequency information is therefore useful in disambiguation.

We have arrived at a generalization of the dictionary used for disambiguation in handwriting recognition: we have constructed here, with symbols (and,

in Section 6, with expressions) a *weighted dictionary*. This structure carries information about the vocabulary of potential results, together with empirically determined weights.

Figure 3 shows the most frequently occurring identifiers for the same classifications after excluding the 10 identifiers that appear most frequently in all classifications together. We see these lists are less similar than those of Figure 2. We might use this information to aid in automatic document classification, together with word-frequency and citation analysis. Information such as this could also be used by an interactive system as a heuristic aid to determine the mathematical area in which a user is working.

Figure 4 shows, for the same classifications, the number of occurrences of identifier symbols, with the symbols ordered from most frequent to least frequent. While this will obviously be a monotonically decreasing curve, it is remarkable the degree of similarity in the shapes of these curves. So, we observe that although *which* symbols are used most varies quite a bit from mathematical area to area, the distribution of use of symbols is remarkably similar.

Although, for space reasons, we have presented here the tabular results and graphs for only three classifications, and for the aggregate, the overall picture is similar for the other classifications.

All			03			11			35		
Ucode	**Op.**	**Freq.**	**Ucode**	**Op.**	**Freq.**	**Ucode**	**Op.**	**Freq.**	**Ucode**	**Op.**	**Freq.**
0029	)	83,542	0029	)	74,308	002C	,	28,836	0029	)	83,792
0028	(	83,397	0028	(	74,246	006E	n	27,743	0028	(	83,478
003D	=	34,773	003D	=	31,706	0070	p	19,216	002D	−	37,843
002D	−	31,376	2061		30,003	006B	k	18,228	2061		28,266
2061		27,861	2208	∋	20,048	0078	x	16,828	003D	=	28,225
002B	+	21,458	002D	−	15,808	0069	i	16,735	002B	+	26,642
2208	∋	11,873	002B	+	15,649	0061	a	12,064	2223	|	25,628
2223	|	10,942	005B	[	12,135	006D	m	11,272	2225	∥	16,608
002A	∗	7,884	005D	]	11,981	0064	d	10,634	2208	∋	10,476
005B	[	7,774	2223	|	9,459	0071	q	10,393	2264	≤	9,440
005D	]	7,619	002A	∗	8,536	0073	s	10,164	2202	∂	7,867
2192	→	6,427	007B	{	7,649	006A	∑	10,086	002F	/	7,096
002F	/	6,328	007D	}	7,453	0072	r	9,391	221E	∞	6,405
2264	≤	5,436	003C	<	7,378	0074	t	9,371	222B	∫	6,333
007B	{	4,839	02C9	¯	6,717	0047	G	9,355	005B	[	5,770
007D	}	4,604	2192	→	6,343	0058	X	9,314	005D	]	5,470
02DC	˜	4,559	2264	≤	6,310	0041	A	9,110	02DC	˜	5,336
2225	∥	3,931	002F	/	3,807	004B	K	9,014	003C	<	4,492
2297	⊗	3,840	2026	…	3,512	0066	f	8,643	2207	∇	4,201
2211	∑	3,669	222A	∪	3,293	0046	F	7,879	003E	>	4,165

Fig. 5. The most frequent operators (per million) in all classifications (All), Logic (03), Number Theory (11) and Partial Differential Equations (35).
The Unicode point 2061 is the invisible "ApplyFunction" operator.

03		
Ucode	**Op.**	**Freq.**
007B	$\{$	7,649
007D	$\}$	7,453
003C	$<$	7,378
02C9	$\bar{}$	6,717
2264	$\leq$	6,310
002F	$/$	3,807
2026	$\ldots$	3,512
222A	$\cup$	3,293
2229	$\cap$	3,249
2286	$\supset$	3,209
003E	$\langle$	3,067
2329	$\rangle$	2,830
232A	$\cdots$	2,758
22EF	$\tilde{}$	2,546
02DC	$\sim$	2,454
00D7	$\times$	2,426
2218	$\circ$	2,315
00AF	$\geq$	2,190

11		
Ucode	**Op.**	**Freq.**
002F	$/$	11,056
2211	$\sum$	5,539
2264	$\leq$	5,377
007B	$\{$	4,423
007D	$\}$	4,134
00AF	$\bar{}$	4,040
221E	∞	4,018
003E	$>$	3,551
22EF	$\cdots$	3,394
02DC	$\tilde{}$	3,353
2265	$\geq$	3,286
2113	ℓ	3,021
003C	$<$	2,788
00D7	$\times$	2,786
2297	$\otimes$	2,434
02C9	$\bar{}$	2,403
2026	$\ldots$	2,150
22C5	$\cdot$	2,122

35		
Ucode	**Op.**	**Freq.**
2225	$\parallel$	16,608
2264	$\leq$	9,440
2202	∂	7,867
002F	$/$	7,096
221E	∞	6,405
222B	$\int$	6,333
02DC	$\tilde{}$	5,336
003C	$<$	4,492
2207	∇	4,201
003E	$>$	4,165
007B	$\{$	3,542
22C5	$\cdot$	3,459
2211	$\sum$	3,384
007D	$\}$	3,155
2265	$\geq$	3,148
00AF	$\bar{}$	2,653
02C9	$\bar{}$	2,552
02C6	$\hat{}$	2,230

Fig. 6. Most frequent operators (per million) in Logic (03), Number Theory (11) and Partial Differential Equations (35), excluding the 12 most frequent from all categories.

5 Operators

An analogous analysis to that for identifiers was performed for operator symbols. Figure 5 shows the most frequently occurring operators for the same classifications as before and Figure 6 shows the most frequently occurring operators, excluding the 12 most common ones from all classifications taken together. Figure 7 shows the count of operator symbols, sorted from most to least popular.

We note that, again, the shape of the operator symbol distribution is similar among categories, even though it is different operators that are occurring most frequently. This shape is also quite different from the distribution for identifiers: generally, a few operators are used very frequently.

6 Expressions

We have performed a similar analysis for non-trivial subexpressions, counting the number of times each distinct subexpression occurs in each subject classification. The analysis of the results is more complex, however.

An large subexpression occurs a certain proportion of the time is more significant than an smaller subexpression that occurs with the same frequency, for two reasons. The first reason is that, in absolute terms, there are fewer subexpressions of the large size occurring. The second reason is that there are exponentially more different potential expressions of the larger size.

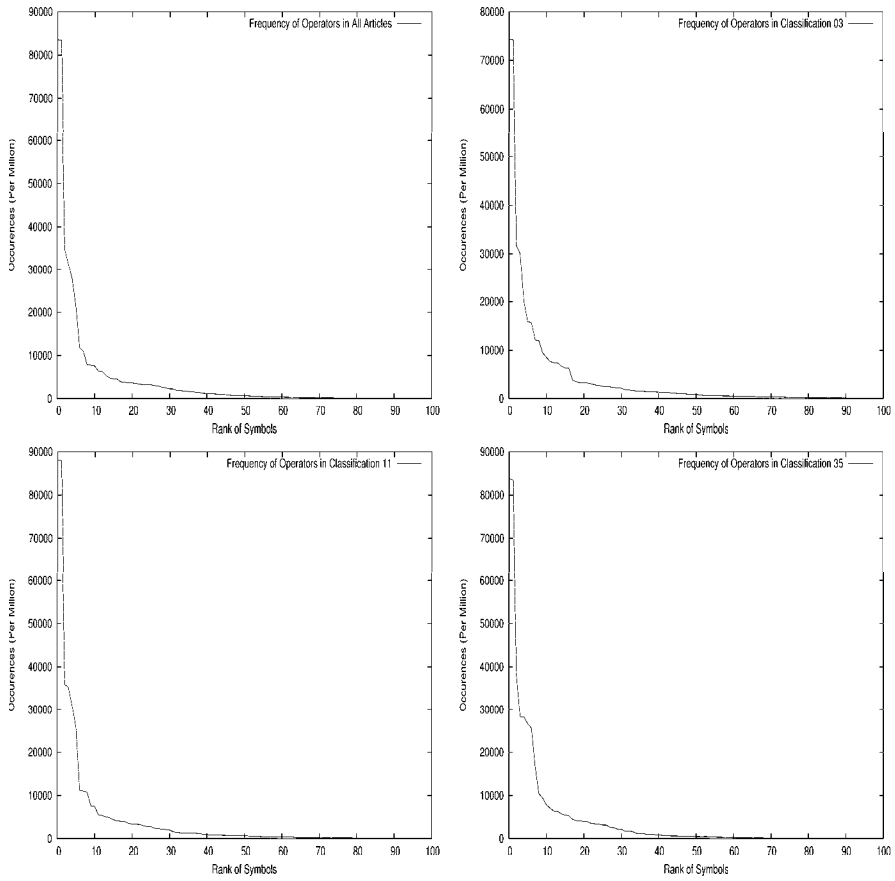

Fig. 7. Most frequent operators in all expressions (upper left), Logic (upper right), Number Theory (lower left), and Partial Differential Equations (lower right).
The horizontal axis gives the symbol (from most to least frequent), and the vertical axis gives the number of occurrences per million symbols in the classification.

With the idea that the size of an expression should be part of determining the significance of its occurrences, we have analyzed each subject classification for the number of expressions, and number of *distinct* expressions, according to expression size. The results for subject classifications 03, 11 and 35 are shown in Figure 8.

We observe two phenomena: First, as expected, the number of expressions occurring decreases as size increases. There are many more small expressions than large expressions. Secondly, we note that the number of distinct expressions decreases more slowly, by size, than the number of expressions. For example, in the classification 35, Partial Differential Equations, there are about 30 expressions of size 2 occurring for each distinct subexpression of size 2. This ratio decreases

03		
Sz	**#**	**distinct**
all	50,933,843	138,136
2	5,151,583	13,439
3	3,113,613	14,183
4	1,703,762	14,276
5	1,294,706	13,631
6	759,075	10,035
7	692,797	9,966
8	422,608	7,094
9	372,049	6,424
10	248,146	4,635
11	235,781	4,515
12	166,687	3,259
13	163,029	3,211
14	117,391	2,491
15	115,599	2,542

11		
Sz	**#**	**distinct**
all	14,293,554	1,362,135
2	1,396,996	65,326
3	887,089	110,311
4	483,089	124,503
5	375,023	130,808
6	220,984	107,670
7	201,022	107,281
8	124,985	78,119
9	108,603	71,658
10	73,020	51,854
11	68,509	49,873
12	49,342	37,912
13	46,860	36,322
14	34,597	28,169
15	33,367	27,404

35		
Sz	**#**	**distinct**
all	9,613,172	802,767
2	924,821	30,670
3	614,469	53,193
4	325,538	59,519
5	238,749	63,393
6	149,664	55,030
7	127,204	54,382
8	86,149	42,599
9	72,703	38,763
10	50,973	30,237
11	44,671	27,931
12	33,966	22,665
13	32,424	21,998
14	24,219	17,371
15	22,997	16,793

Fig. 8. Number of subexpressions and of distinct subexpressions by classification and by subexpression size

#	Expression
19717	-1
15657	L^2
7903	dx
5661	t_0
4837	u_0
4752	x_0
4462	∂_t
4459	ij
4095	tx
3874	dt

#	Expression
4053	(t, x)
3399	(x, t)
2230	(x, y)
2229	$[0, T]$
1985	$-1/2$
1727	(x, ξ)
1547	$[0, 1]$
1374	(x_0)
1327	(t_0)
1206	(R^n)

#	Expression
1197	$\|x - y\|$
1163	$(n - 1)$
920	$(t - s)$
799	$(n - 2)$
733	$u(t)$
569	$(t, \cdot)$
508	$(x - y)$
499	$\frac{n-2}{2}$
496	$\|\nabla u\|^2$
441	$\Omega_0; R^3$

Fig. 9. Most frequent subexpressions of size 2 and of size 4-5 in subject classification classification Partial Differential Equations (35).

steadily as the size increases: there are about 2 expressions of size 8 occurring for each distinct expression of size 8, and the ratio is less than 1.5 for size 15.

This analysis provides a weighted dictionary for each subject classification, providing the frequency that expressions occur in each subject classification. Space limitations preclude giving a detailed accounting of the particular expressions which occur most frequently in each classification, but we give a sample from the classification 35, Partial Differential Equations. These are shown in Figure 9. More details are available in [7]

The information in this weighted dictionary may be used directly by applications, or may be used for further analysis. Two such directions of further analysis are deriving expression patterns, and deriving common writing sequences.

Expression Patterns

We note that very similar subexpressions may occur frequently, for example $\sqrt{A^2+B^2}$ and $\sqrt{x^2+y^2}$. While it is possible to maintain a weighted dictionary keeping track of both of these expressions, it would be more desirable to determine that $\sqrt{\alpha^2+\beta^2}$ was a frequently occurring pattern, with suitable choices of α and β.

"Antiunification" provides an elegant framework to define such patterns. Antiunification is a process dual to unification. Rather than taking expressions and determining the most general expression to which they all can be specialized, antiunification takes a number of instance expressions and finds the least general expression which may be specialized to each instance expression. The syntactic form of antiunification has been studied since the 1970s [6].

We may determine the set of patterns from a weighted dictionary by considering all pairs of expressions. Each pair will give an antiunifier. We then consider all pairs of antiunifiers with expressions from the dictionary. These may give more antiunifiers, which are added to the set of antiunifiers. We continue to consider pairs of antiunifiers with expressions until no new antiunifiers are generated. Since antiunification is associative, this generates a complete set of antiunifiers for the dictionary. For each antiunification, we may use the one pass algorithm of [9].

We may associate weights with these patterns simply: for each antiunifier, attempt a unification with each expression in the weighted dictionary. Then the weight of the antiunifier is the sum of the weights of the expressions with which it unifies. We note that since we are interested in syntactic expressions, this entire process of antiunification and unification is syntactic. An empirically derived, weighted dictionary of antiunifiers would provide an interesting measure to select among possibilities for "simplified" forms in a computer algebra system.

Tree-Order Symbol Seqences

The second direction we wish to discuss for deriving expression patterns is the use of ordered tree traversals. We examine this in support of mathematical handwriting recognition.

For each type of tree node, we define a traversal order corresponding to the most common writing order. For example, with

$$\sum_{i=0}^{\infty} i^2$$

the summation sign is usually written first, followed by the equation $i = 0$, then ∞, and finally i^2. Ideally the information on writing order for each node type should be determined with user experiments. Without these experiments, it is still possible to have writer-specific traversal order.

Given one or more traversal orders for each node type, we may then examine the weighted dictionary of expressions, traversing each expression, to determine

Markov chains for symbol sequences. If the expression $\sum_{i=0} \ldots$ occurs twice as frequently as $\sum_{j=0} \ldots$, then the symbol sequence $\langle \Sigma, i \rangle$ gets twice the weight of $\langle \Sigma, j \rangle$. If there is not a unique traversal order for a node type, then the alternatives may be weighted.

One can easily imagine additional uses of this kind of empirical data on expression frequency.

7 Conclusions

We have proposed the idea of empirical analysis of mathematical literature as a new technique to be used in the design of sophisticated mathematical software. This is a break from the tradition of system designers using their own preferences or prejudices in determining which forms of expressions will be deemed most preferable by their systems.

We have taken presented an approach to performing empirical analysis of a body of mathematical literature. We have developed a suite of tools to convert raw TEX source to well-formed MathML, and to build weighted dictionaries of symbols and expressions.

We have made an analysis of all articles from `arXiv.org` since the new MSC 2000 subject classification. From this, we have observed that the use of mathematical symbols varies considerably from area to area and have produced usage frequency tables for all MSC 2000 classification areas. We have observed that, while the specifics of *which* symbols are most used varies from area to area, the overall *distribution* of symbol use is very similar between areas. This is true both for symbols used as identifiers (function names and arguments), and as operators. We have also analyzed the collection of subexpressions present in the `arXiv.org` data. As well as developing a weighted dictionary for each classification area, we have observed some general properties of the frequency of distinct expressions.

Beyond these practical experiments, we have explored the potential use of information derived from symbol and expression weighted dictionaries. These have included particular applications to computer algebra, mathematical handwriting recognition and document analysis. We have also shown how weighted expression dictionaries may be used to determine further useful information, including weighted pattern dictionaries (by antiunification) and Markov chains for symbols in writing-order traversal of expression trees.

The applicability of these results depends on how representative the empirical data is. It is likely that different tables would be obtained from high-school mathematics texts, for example. Therefore, the overall approach we have taken is just as important as the specific results for this particular mathematical database.

We are excited and hopeful that the use of empirically gained knowledge may make mathematical software systems more powerful and more natural to use.

References

1. David Carlisle, Patrick Ion, Robert Miner, Nico Poppelier, Editors. Mathematical Markup Language (MathML) Version 2.0 (Second Edition). W3C Recommendation. `http://www.w3.org/TR/2003/REC-MathML2-20031021/`. October 21, 2003.
2. ArXiv e-Print Archive. `http://xxx.lanl.gov`
3. Mathematical Subject Classification (2000). American Mathematical Society. `http://www.ams.org/msc`
4. The Hermes Project. `http://alphaserv3.aei.mpg.de/hermes`
5. Ontario Research Centre for Computer Algebra. Online TeX to MathML translator. `http://www.orcca.on.ca/MathML/texmml/textomml.html` (2002)
6. Gordon D. Plotkin. A Note on Inductive Generalization. Machine Intelligence 5 153–163 (1970).
7. Clare So. An Analysis of Mathematical Expressions Used in Practice. MSc. Thesis. University of Western Ontario. (2005)
8. Stephen M. Watt. Implicit Mathematical Semantics in Conversion between TEX and MathML, TUGBoat, Vol 23, No 1 (2002)
9. Stephen M. Watt, Clare So and Cosmin Oancea. Generalization in Maple (accepted), Maple Conference 2005, Maplesoft.