A Solution to Mathematics Parsing

Martin Proulx

April 29, 1996

1 Introduction

1.1 What the problem is

We need to derive the mathematical meaning from a collection of typeset symbols.

1.2 Why?

TILU: Automate the entry of large, old, and unavaible electronically book of integrals.

Multivalent documents: Once a mathematical expression is recognized (as being mathematics rather than text or graphics) and located, we would then like to parse it or derive the TEX, LISP or ??? equivalent.

1.3 The possible solutions

A grammar based approach:

This would be the ideal, but has some major drawbacks. First, normal grammars do not provide us any ways to describe the 2D environments that mathematics really are. If we take a grammar based approach, we also have to come up with our own 2D extensions in the grammar for things such as "in the exponent of", "above", etc...

Then, even if we can define a grammar, is that of any help? Not really, since there aren't any programs that will let us parse the mathematics in any reasonneable amount of time. Which brings us to the other solution.

Writing a specific program:

This is what I opted for after first trying to define a grammar for the problems we had to deal with. Defining a grammar was really helpful in that ended up seeing what was common between expressions, and what wasn't. After having a good idea of the overall picture, I could start writing the parser.

1.4 Bigger doens't mean harder!

Before going on, it's also important to realize that the problem doesn't reside in the size or mathematical complexity of the equations. The real problem is figuring out the spatial layout of the

symbols, and assign to those symbols with this layout a mathematical meaning.

As an example of this, consider:

$$\int \frac{dx}{x-1} = \left(\frac{x^p}{\cos p\pi}\right)$$

and:

$$\int \frac{x^p - x^q}{x-1}\frac{dx}{x+r} = \frac{\pi}{1+r}\left(\frac{r^p - \cos p\pi}{\sin p\pi} - \frac{r^q - \cos q\pi}{\sin q\pi}\right)$$

Those two equations are just as hard, because they both contain the following: integrals, equal sign, exponents, parentheses and function (sin, cos).

2 My solution

2.1 The assumptions and the starting point

My parser assumes that perfect recognition of the symbols have been made, and that all those symbols have been collected by a lexical analyser. I.E. The parser works with tokens such as "sin" or "quotient" and "minus" rather than with "s", "i", "n" and two "hlines" of different lengths.

2.2 The concepts

The parser I implemented is an operator based recursive descent parser. So at every step, it will look at the subset of the symbols it has looking for certain mathematical operators, take a stand on how it is appropriate to split this subset of the equation, and recurse on the new smaller subsets. There's a visual example of recursion in figure 1.

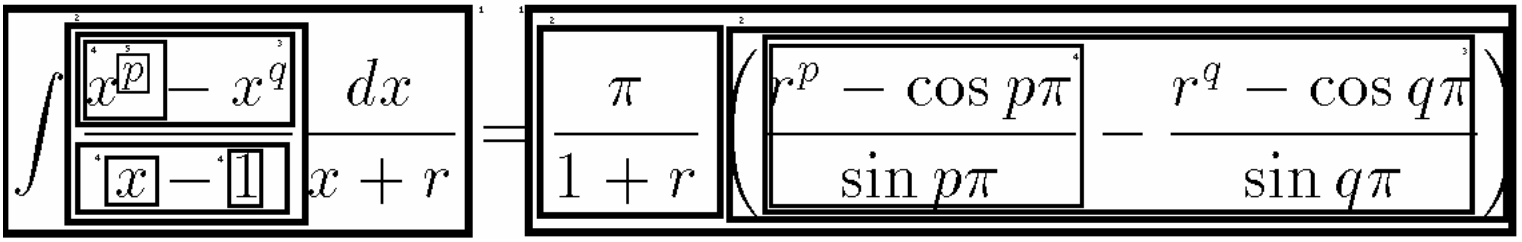

Figure 1: Visual example of recursion

The problem in doing so is in figuring out where to look for those operators, and which ones to look for. In order to differentiate the top-level expression from sub-expressions (like exponents, or expressions above or below divide bars), I gather what I call the main-line of the expression.

The mainline concept turns out to be the most useful idea of the whole technique that I'm using. Rather than to deal with the baselines, I deal with what I call the "mainlines". Look at figure 2.

This makes it terribly easy to figure out if some symbols are in the exponent or subscript of another symbol. It also clearly show you where the top level of your expression is, and where to look for key symbols.

How do I find them? It's simply the vertical middle of the symbol for most, and you need to adjust this slightly for characters with ascenders and descenders.

How do I find the mainline of the top-expression? It's nearly always the mainline of the left-most character, but for some special cases. Like the one in figure 3.

2.3 The different operators, by decreasing order of priority

"," and ";": those are used to split expressions one of another, usually found inside parentheses and separating expressions that are arguments to a function. we need to look for them on the mainline, and then recurse.

"=": We need to look for this one on the mainline, and then recurse accordingly.

"-", "+-", "-+": Used as unary-operators. Those will be the leftmost symbol of the expression.

"+","-" : We need to look for this one on the mainline, and then recurse accordingly.

"/" This is different from the horizontal bar, since it seems from its usage to have a different priority. I.E What is $(1/2 - \pi)$? Is it a half minus pi or 1 over (2 minus pi)? I implemented the first possibility.

All the rest are looked for in the leftmost symbol.

divide bar: This is the long horizontal bar with an expression above and below as well. We will then recurse above, below, and on the right of it.

integral,sum: Those two need some constant special attention, regarding where to find their ranges. We then recurse into everything but the symbol, the ranges and the "dx" for integrals.

root: We get the possible exponent the exact same way as we would for normal variables. Then recurse inside the root symbol, and on its right.

functions: Need to find where the arguments are, and recurse accordingly.
Opening parenthesis: Need to find the matching parenthesis, and recurse both into, and on the right of the matching parenthesis.

Variable: Multiply it and its exponent/subscripts against whatever is on the right.

Special case: factorials "!" I do not look for factorials at all during the parsing. I simply treat them as any other symbol, and when I'm done parsing, I have a little tail-end program that will unwind things such as (* (expression) fact) into (fact (expression)).

3 Conclusion

3.1 What I've parsed succesfully so far

Look at comments on integral formulas, OCR, and parsing.

3.2 The loose ends

Dependance upon the correct typesetting. The parser couldn't recognize correctly an expression if it can't derive good mainline infos, which are dependent upon typesetting.

Assuming perfect input may not be feasible, see figure **??**.

Notice that the integral sign and the b are touching. The exponents are hardly readeable. Will the OCR be able to break the pieces as we'd like and be able to recognize those exponents?

Solutions for this. A stochastic parser, where the parser knows the different probabilities for possible meanings of not clearly recognized symbols. The parser would then return the expression if belives to be the more plausible according to other known factors.

3.3 The open ends

Partial recognitions. Right now, because of the assumptions, the parser will totally fail if it encounter an expression that is not correctly typeset. The parser should be expanded to be able to skip sections and say it couldn't figure out a certain part, but still parse the sctions it can. Examples of not correctly typeset?

More knowledge. Expand the parser to deal with matrices. Also give it more power in recognizing functions versus simple variables.

Parsing multiple-lines expressions.

Keeping typesetting information.

The whole problem needs a solid lexical analyser.

$$X_{\nu}^{\pm} = \frac{b^{-\alpha/r}}{2} \sum_{k=0}^{\infty} \frac{1}{k!} [B_k(\nu) + B_k(-\nu)] \left(\pm \frac{2c}{b^{1/r}} \right)^k$$

Figure 2: Visual example of mainlines

$$\int_{-q}^{q+1} \sum_{x=1}^{q} x \quad dx$$

Figure 3: When the physical leftmost isn't the right one

$$\textbf{13.} \quad \int_b^u \frac{x^2\,dx}{\sqrt{(a^2-x^2)^3(x^2-b^2)}} = \frac{1}{a(a^2-b^2)} \left\{ b^2 F(\varkappa, q) - a^2 E(\varkappa, q) + \frac{a^2}{u} \sqrt{\frac{u^2-b^2}{a^2-u^2}} \right\}$$

$$[a > u > b > 0]. \qquad \textbf{BY (217.06)}$$

Figure 4: A real world example