A Search System for Tree-Structrued Mathematical Formula in *LaTeX* Format

Wei Zhong, Hui Fang
Dept. of Electrical and Computer Engineering
University of Delaware
Newark, DE USA
{zhongwei, hfang}@udel.edu

ABSTRACT

The special characteristic of mathematical language, compared with normal text content, makes mainstream retrieval models (e.g. "bag of words" model) deficient to provide good result in cases where query is in mathematical language. In this paper, we address the difficulties of similarity identification of mathematical contents and present a search system specifically for mathematical contents, more precisely, a system to tokenize mathematical content in LaTeX into formulas, to transform a tree-structured formula to what we call "branch words", and to score the relevance degree between "branch words". Using this system, we show the potential and possibility to tackle the problems in mathematical searching.

Categories and Subject Descriptors

H.3 [**Information Search and Retrieval**]: Miscellaneous

General Terms

Algorithms

Keywords

mathematical searching, language processing, search engine

1. INTRODUCTION

Unlike normal text content, mathematical language, by its nature, has many differences from normal text content. First, one mathematical formula may use different notations and thus has different representations.[1] Second, two semantically different mathematical document can contain the same number of terms.[2] Also, the order of terms in math language sometimes matters but can be mutable in other cases, which implies it is impractical to uniformly apply one single language model to mathematical content.

[1]An example, $a^2 + b^2 = c^2$ can be described as $x^2 + y^2 = z^2$ as well.
[2]Consider $ax + (b + c)$ and $(a + b)x + c$.

Currently existing and working mathematical searching systems that we can find online, primarily are *WolfraAlpha*[3], *(uni)quation*[4] and *MWS*[5]. *WolfraAlpha* mainly focus on computation and evaluation of mathematical equation input rather than searching the similar equation. *(uni)quation* is able to solve the problems addressed above in this section but it is not under development and the model and method it uses is currently unclear.[6] *WMS* uses Term indexing[1, 2] which we are not using here, to discriminate the mathematical structures.

Our system tries a different approach to make use of the structure of mathematical formula to solve the problems addressed above. It uses an efficient way to parse and tokenize mathematic formula, to transform a tree-structured formula to a more comparable structure, and also gives the method to score and rank results.

2. SYSTEM DESCRIPTION

Our system has a WEB front-end to accept user input in LaTeX format and pass it to the back-end. The back-end, mainly consists of a parser and a search program. The function of parser is to do tokenization and tree construction as well as storing output structure into our collection. The search program will then compare the query and document and evaluate the similarity of them, and give the final ranking through output file for the WEB front-end `CGI program`[7] to read. We input mathematical content in LaTeX as document to our parser by either manually selecting or a crawler script specifically targeting at mathematical content website *Mathematics Stack Exchange*[8].

2.1 Tokenization and Tree Construction

We choose to tokenize a subset of mathematic related LaTeX language, we use `Lex/Yacc` tools to tokenize the LaTex Language and construct a "tree" for each equation. The LALR parser generator of `Yacc` can handle language efficiently in guaranteed linear time[3]. The grammar we use will parse mathematical content into different classes of tokens including variables, different basic mathematical operators, equal class, times class, fraction class and square root. And we

[3]http://www.wolframalpha.com/
[4]http://uniquation.com/
[5]http://search.mathweb.org/
[6]http://rystsov.info/2009/05/01/uniquation.html
[7]Common Gateway Interface, here we use as a way for Apache web server to interact with external programs.
[8]http://math.stackexchange.com/

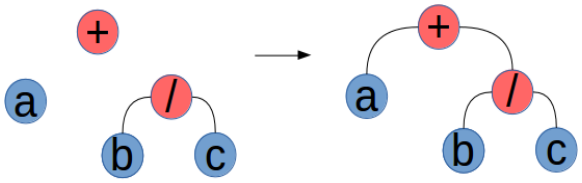

Figure 1: Example of Sub-tree generation for the addition grammar.

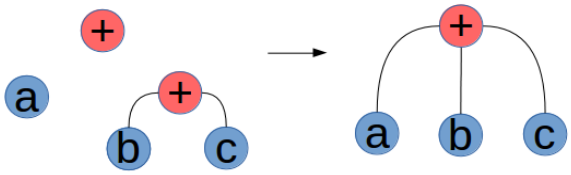

Figure 2: Cases where commutative property applies.

choose to omit undefined control sequence for the sake of robustness.

When a grammar is reduced, the tokens is converted to a tree node directly or by attaching sub-trees which is reduced previously to the new root. In this way, we will finally get a tree structured representation.

Some operations may have commutative property, in these cases[9], all the sons in two adjacent levels will be attached to the same root.

2.2 Extraction of Branch words

After constructing a tree, A "branch word" is extracted by taking tokens from the leaves to the root of a tree in order. Branch words are used to be compared with those of other trees.

[9]In our system, the cases where we apply commutative property include *addition* and *multiplication* operations.

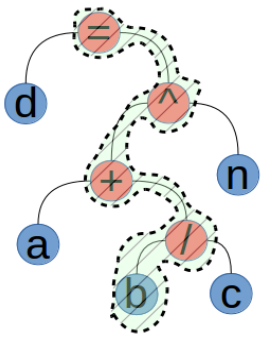

Figure 3: Illustration of One Branch Word for formula $(a + \frac{b}{c})^n = d$.

We notice that one math equation can use different symbol sets, so we choose not to distinguish the leaves' actual symbol in the branch word. We also notice the branch word is not enough to distinguish trees, an example would be the equation $(a + b + c) \times (d + e)$ and $(a + b) \times (c + d + e)$, they have the same branch words yet have different semantic meaning in mathematical language. To avoid this flaw we further introduce the weight for any node n_i in a branch word, defined by the sum of that of its successors, which is given by:

$$w(n_i) = \sum_{n} 1\,, \qquad n \in \{n_i \cup succ(n_i)\}$$

2.3 Indexing and Storage

The storage of mathematical formulas in our system, contains all the branch words from each tree as well as the weight information of each node in branch words. To enable efficient retrieval, branch words are stored in file system where the path is named by token names of a branch word (with weight information) in order[10]. The path where a branch word resides also store a "posting" file recording all the documents in the collection that contain that branch word.

2.4 Comparing and Scoring

In our system, comparing two pieces of mathematical content is essentially to compare all the branch words from one content with those of the other, calculate the similarity degree between two branch words. In terms of search system which rank-orders the documents matching a query, a score with respect to the query for each matching document is computed by sum the similarity degree for each related document in the document collection, then rank all the related documents using the sum score.

For the detailed caculation, we put some notations here: Let m be the number of continuous weight matches between two branches from the beginning of the branch word. For one branch word b, let n_b be number of same branch word from document that extracts b, l_b be the length of branch word b. Then for all query branch word i and all document branch word j. The formulas we use to calculate both difference degree D between two branch words and the score S_k for a related document k are:

$$D_{i,j} = \alpha \cdot \min(n_i, n_j) \cdot \frac{m}{l_i} + \frac{1}{|n_i - n_j| + 1} \cdot \frac{m}{\max(l_i, l_j)} \quad (1)$$

$$S_k = \sum_i \sum_j D_{i,j} \quad (2)$$

Where α is a large constant number to prioritize the first term in (1).

2.5 WEB front-end

The WEB interface contains a homepage for user to input query in an input box, just like a normal search engine does. The difference is, our homepage accepts LaTeX as input and has a render preview for math equations. The WEB interface uses CGI program as a middle layer to both get user query (using *libcurl* library to unescape the URL encoding)

[10]One example can be `./collection/var/frac/add`

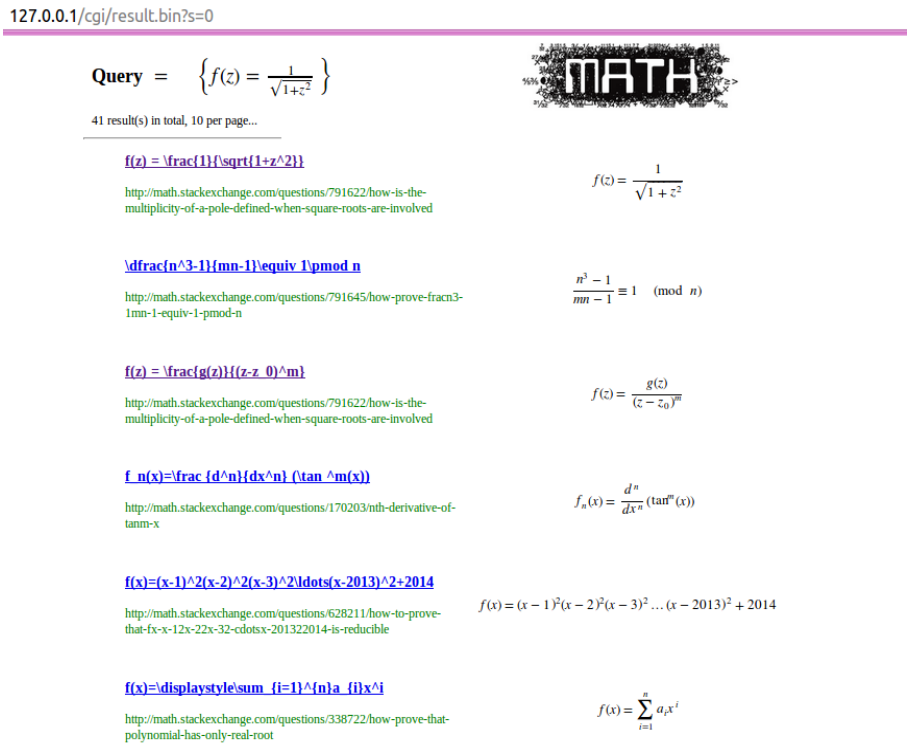

Figure 4: WEB interface for Our Prototype System

and generate ranking page in HTML (by writing to the standard output) from the output file written by back-end program.

3. FUTURE WORK

Our current system does not distinguish variables and constant tokens from each other. Although this is good in cases a^2+b^2 and x^2+y^2 is treated as the same, but it is obviously not reasonable to let $n+\frac{1}{n}$ equal with $a+\frac{1}{b}$. Future work may involve unification algorithm[4] to better evaluate the relevance between formula symbols. Second, the resulting collection with input data from our crawler is not as good as that with the input data we manually choose. Part of the issue arises because the ambiguity of high-level spaces[5]. Erroneous tight-binding spaces and failure in interpretation will result in blank token in our system, which may be deduced by our parser as *multiplication* operation. Moreover, statistical evaluation needs to be done on the effectiveness of our search results. But above all, the most important goal of future work is to explore the huge potential and possibilities of math-aware searching.

4. CONCLUSIONS

In this paper, we introduce and present an experimental mathematical formula search system, also demonstrates the possibility of our approach, which tries to utilize the structure of math formula to provide better search result for mathematical content. Our prototype is able to return relevant result in cases where the query uses a different set of symbol notations from the document, and a change-of-order for symbols that has mutable property. While there is a lot of future work ahead, the rich possibilities and potential for math-aware searching remain inspiring for us.

5. REFERENCES

[1] W. W. McCune. Experiments with discrimination-tree indexing and path indexing for term retrieval. *Journal of Automated Reasoning*, 9(2), 1992.

[2] Stickel. The path-indexing method for indexing terms. In *Technical Report*, page 473. Artificial Intelligence Center, SRI International, October 1989.

[3] D. E. Knuth. On the translation of languages from left to right. *Information and Control*, 8(6), 1965.

[4] F. Baader and W. Snyder. Unification theory. *Handbook of Automated Reasoning*, I:447–533, 2001.

[5] Richard J. Fateman and Eylon Caspi. Parsing tex into mathematics. August 1989.